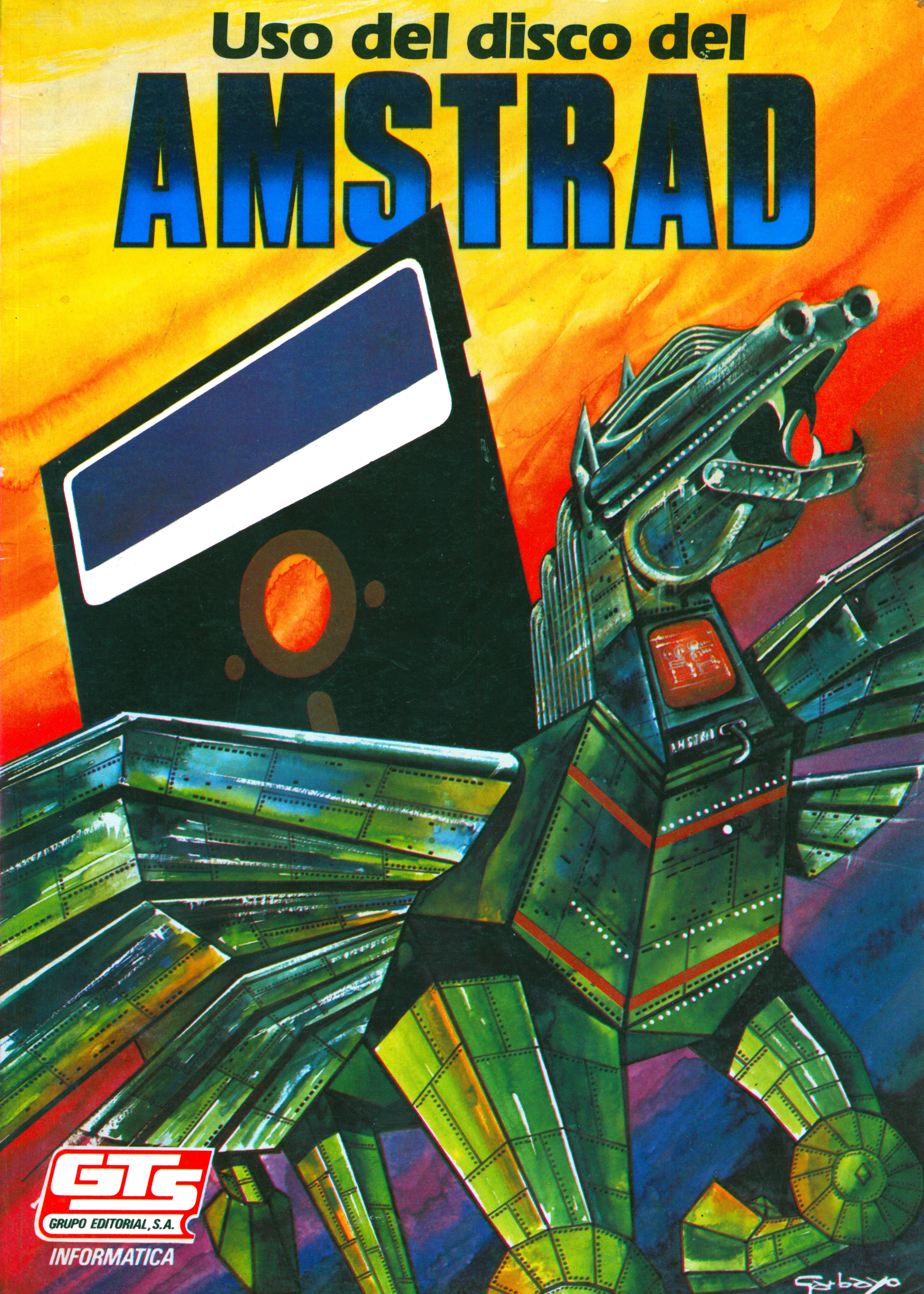
Uso del disco del AMSTRAD



USO

DEL

DISCO

DEL

AMSTRAD

POR

C. LONGHI

PROLOGO

Los Ordenadores Personales no son unas unidades simples, sino un sistema de dispositivos y programas los cuales haran de su equipo una herramienta muy practica tanto para su entretenimiento como para su negocio o estudio.

El Disco del AMSTRAD es un dispositivo el cual le da mas flexibilidad a su Ordenador, dado que en el se podran encontrar un Sistema Operativo CP/M (nombre registrado por Digital Research), el AMSDOS (nombre registrado por Amstrad Ltd.) que es el sistema del Disco, y el leguaje de Dr. LOGO.

Aunque Vd. crea que el CP/M es un programa muy complejo de Ordenador, lo podra aprender a utilizarlo sin ningun problema ni experiencia previa dado que como vera por este libro, su manejo es muy simple. Asi mismo el leguaje Dr. LOGO le resultara muy interesante dado su potente interpretador y su facilidad en programacion.

El poder tener acceso al Sistema CP/M es muy practico, pues en el mercado de la informatica existen una serie de programas de utilidad que le podran resolver sus problemas en el uso de su Ordenador, tales como el MBASIC que es un lenguaje de Basic muy potente, el CBASIC el cual es un compilador de Basic, un WORDSTAR que es un procesador de textos muy popular dentro del CP/M y la clasica base de datos DBASE. Estos programas y otros mas, los podra Vd. usar en su equipo sin mas problemas, dado que el CP/M es un sistema estandar para todos los conputadores.

Quiero agradecer la gran ayuda que me ha prestado la firma PEEK & POKE, en la calle Genova 11 de Madrid, por la asistencia en la elaboracion de este libro, pudiendo usar los Ordenadores AMSTRAD en la verificacion de los diferentes capitulos.

CONTENIDO DE ESTA OBRA

CAPITULO 1

1.1 INTRODUCCION AL USO DEL DISCO

En este libro, no tocaremos lo que se relaciona con la instalacion del Disco (DDI-1, modelos 664 y 6128) dado que en el Manual que Vd. tiene, viene explicado a la perfeccion, por lo que pasaremos directamente al uso del mismo y sus aplicaciones de los diferentes lenguajes.

Los Discos flexibles de esta unidad, tienen una dimension de tres pulgadas, y pueden ser grabados por las dos caras, obteniendo una capacidad total de 360 K Bytes (180 K por cada lado) con lo que le da una disponibilidad de espacio mayor que la de la Cassette, y un acceso mas rapido a sus ficheros.

Otra de las ventajas de estos Discos, es su consistencia, dado que estan protejidos por una carcasa rijida de plastico con un cierre protector, cubriendo el area donde las cabezas del lector deben pasar, impidiendo un dano accidental.

Asi mismo, en la parte izquierda de cada cara del Disco, se podra observar una flecha indicando un pequeno hueco con un cierre, el cual esta destinado a la "Proteccion de escritura" evitando de este modo la sobre escritura de programas importantes (lenguajes, juegos, utilidades, etc.), en su manual de usuario esta explicado el uso correcto de esta ranura de proteccion.

La carga de los programas disponibles en Disco para este sistema, son cargados directamente en memoria desde el mismo Diskette, mucho mas eapido y seguro que desde la cinta de Cassette, pero sin embargo, hay que recordar unas reglas importantisimas relacionadas con los nombres de los ficheros que van a ser cargados en memoria.

El comando, normalmente usado en cinta de Cassette, "RUN" para cargar programas que han de ser ejecutados, no es valido en las operaciones de Disco, dado que es necesario el poner el nombre del fichero (programa) cuando se trabaja en esta modalidad, constando este nombre de dos partes o campos separados por un punto (.). El primer campo podra tener hasta 8 caracteres, siendo habitual el nombre del programa, y el segundo campo es optativamente opcional, teniendo una relacion con el tipo de programa (ej. BAS por Basic o BIN por Binario). Hay que recalcar, que el nombre del fichero no tiene que tener espacios en blanco o signos de puntuacion en ninguno de los dos campos, pudiendose usar, tanto letras mayusculas como minusculas en los diferentes ficheros de los comandos de AMSDOS y CP/M.

1.2 CARGA DE PROGRAMAS RESIDENTES EN DISCO

Para que Vd. se ponga en contacto con su unidad de Disco, empezaremos con la operacion de cargar un programa residente en un diskette, y para ello debera teclear lo siguiente:

RUN "NOMBRE" <ENTER> (donde el NOMBRE es el fichero a cargar)

Si se ha insertado correctamente el Disco que contiene los programas, el fichero deseado sera extraido y cargado en memoria en pocos segundos, ejecutandose inmediatamente, pero si por casualidad no se ha insertado el Disco, o se ha equivocado de unidad de Disco, le aparecera en pantalla un mensaje de error como:

```
Drive A: disc missing
Retry, Ignore or Cancel?
```

indicandole la falta de Disco en su undad A y requiriendo si lo intenta de nuevo, lo ignora o lo cancela. Pero si no encuentra el programa en ese Disco, el mensaje de error sera el siguiente:

```
NOMBRE - not found
```

teniendo que analizar su Disco para ver si el programa deseado se encuentra en el catalogo del mismo o pudiese ser que Vd. se equivocase de nombre de fichero. Otro tipo de error, seria el siguiente:

```
Bad command
```

significando que se ha equivocado en la introduccion del comando, programa escrito incorrectamente, o bien sea por que se ha sobrepasado en el numero de caracteres de alguno de los dos campos, o haber puesto algun espacio en blanco o puntuacion dentro del nombre del fichero. El siguiente error que le puede aparecer seria el de:

```
Type mismatch
```

el cual le indica que ha omitido algo en su comando, como por ejemplo las comillas (").

```
Drive A: read fail
Retry, Ignore or Cancel?
```

Este mensaje de error significa que ha tenido un fallo en el proceso de lectura del Disco A con lo que le le pide que lo intente de nuevo, lo ignore, o lo cancele. Una vez comprobado que se ha introducido el Disco correctamente, pulse la letra R para intentarlo de nuevo, pero si este mensaje continua apareciendo, la causa de este fallo sera que el disco este deteriorado o que no se formateado correctamente.

El error sintactico es muy corriente que le aparezca, dado que hay mucha posibilidad de cometer un error de escritura en el teclado, por lo que le aparecera este mensaje:

```
Syntax error
```

Por ultimo el mensaje de:

Press PLAY then any key:

que su significado es el mismo que cuando se usaba la cinta de Cassette, es por que la undad, o unidades de Disco, o interface del mismo no se han conectado correctamente al Computador, asi como que la unidad no tiene suministro electrico por no haberse enchufado a la red.

Con estos mensajes de ERRORES cubrimos parte de las dificultades que Vd. puede tener para cargar un programa de Disco.

1.3 SYSTEMAS AMSDOS Y CP/M

Cuando su Computador es conectado, el sistema comprueba con una operacion interna, para ver que todos los perifericos esten conectados en sus respectivos zocalos, asi pues, si se detecta que una unidad de Disco esta conectada, interpretara que los comandos destinados hacia la Datarecorder (Cinta de Cassette), los ejecutara enviandolos a la unidad de Disco, actuando sobre el mismo los siguientes comandos:

```
load "nombrefichero"
run "nombrefichero"
save "nombrefichero"
chain "nombrefichero"
chain merge "nombrefichero"
merge "nombrefichero"
openin "nombrefichero"
openout "nombrefichero"
closein
closeout
cat
eof
input #9
line input #9
write #9
list #9
```

Asi pues, dado que se ha conectado la unidad de Disco, el Computador sera regido por el sistema operativo AMSDOS, permitiendo programar el BASIC del Amstrad normalmente, ampliandolo a los comandos para la gestion de ficheros en Disco, denominados "Comandos Externos", no estando disponibles cuando no se tiene conectada la unidad de Disco, dado que estos comandos estan en un circuito de ROM dentro de la unidad del Disco (solo en el modelo 464), estos son identificados dado que van precedidos por el simbolo ¦. (consiguiendolo desde el teclado, presionando simultaneamente la tecla SHIFT y la marcada con @).

Algunos de los comandos mas normales que se emplean son los siguientes:

```
¦a
¦b
¦tape
¦tape.in
¦tape.out
¦disc
¦disc.in
¦disc.out
```

Para indicar al Computador hacia que unidad de Disco se tiene que dirijir, se usan los comandos ña y ñb. Asi por ejemplo si se teclea:

```
¦a
load "NOMBREFICHERO"
```

el Computador se dirijira a la unidad de Disco "a" para buscar el nombre del fichero y cargarlo en memoria. Si por el contrario, se ha omitido el comando ¦a, el Computador se dirijira a la unidad principal (a).

En el caso de que Vd. desease utilizar la cinta Cassette, debera usar el comando "ñtape", el cual le indica al Computador que todas las operaciones de lectura, escritura, etc, se efectuen en la cinta en vez de en el Disco. Si Vd. desea volver al sistema de Disco, solo tendra que teclear el comando "ñdisc".

Otra de las ventajas que tiene este sistema, es que se puede trabajar simultaneamente con Cassette y Disco, por lo que si Vd. tiene un programa en cinta y desea que se guarde en su Disco, lo unico que debe hacer es teclear el comando "¦tape.in", con lo que se le indicara al Computador que el programa sera cargado desde la cinta a la memoria y luego salvado en Disco. De la misma manera, se puede sacar informacion de un Disco y pasarlo a una cinta, por lo que primeramente se debera teclear: "¦disc.in", para contrarrestar el comando anterior de "¦tape.in", y luego "¦tape.out" para indicar al Computador que la cinta de salida es un periferico hacia donde tiene que ir la informacion que en ese momento salga.

Todos estos comandos, los podra ver mejor en los siguientes Capitulos, donde se mostraran ejemplos de duplicidad y copia de programas.



2.1 SISTEMA OPERATIVO CP/M

CP/M es un Sistema Operativo en Disco para microprocesadores producido por la compania Digital Research, y sus siglas significan, "Control Program/Monitor". Existen varias versiones de este tipo de Sistema Operativo para una amplia gama de sistemas, siendo el CP/M-80 el que se puede usar con casi todos, donde su unidad central de tratamiento sea un 8080 o Z80 y con unidades de Disco.

CP/M fue desarrollado en 1973 por el Dr. Gary Kildall, que a su vez era analista consultor de la firma Intel. Su primera version fue escrita para el propio sistema experimental del Dr. Kildall, que incluia una de las primeras unidades de Disco de 8 pulgadas, mostrando las primeras versiones de este Sistema Operativo a la firma para quien trabajaba (Intel), la cual declino comercializar o desarrollar ulteriormente el proyecto, dado que en esa fecha, los microcomputadores eran una rareza y los que tenian no estaban muy seguros de lo que podian hacer con ellos.

Otro elemento importante en la historia de CP/M fue el entusiasmo de sus primeros usuarios, autenticos aficionados que encontraban normalmente problemas insuperables en persecucion de nuevos conocimientos y experiencias. Realmente el CP/M-80 enlazaria cualquier microcomputadora basada en el 8080 o Z80 con cualquier sistema de Disco, y un grupo de aficionados con sistemas entremezclados (mix and match) surgio para probar el producto del Dr. kildall, desarrollando una cantidad de refinamientos y, sobre todo, un grupo fuerte y visible de usuarios.

Entre las herramientas de desarrollo que ayudaron al establecimiento de CP/M como el Sistema Operativo idoneo para microcomputadores, hay que considerar el CBASIC (y su predecesor EBASIC), y diversos programas especiales de ensamblador. De hecho estas herramientas se usaban para escribir programas de aplicacion, tales como libros, programas de inventario, base de datos y programas de tratamiento de texto.

La popularidad del Sistema Operativo CP/M llego a ser parte de un modelo de escalada, CP/M expandia lenguajes de programacion y herramientas de desarrollo que, a su vez, daban lugar a programas de aplicacion. Estos dependientes de CP/M, aumentaban las ventas, conduciendo a un nuevo aumento de programas de desarrollo y asi sucesivamente.

Probablemente Vd. compro este libro porque necesita ejecutar un programa de aplicacion en CP/M, pudiendo ser un simple programa de tratamiento de textos o un sufisticado sistema de contabilidad, pero ambos requieren un conocimiento del CP/M.

Los manuales para CP/M de Digital Research, no fueron escritos para usted, sino para programadores profesionales (esto no quiere decir que Vd. no lo sea), asi pues este libro intenta cubrir la distancia entre esos libros y sus conocimientos sobre Computadoras.

Digital Research esta mejorando constantemente este sistema de CP/M, de forma que cada cierto tiempo suministra nuevas versiones de CP/M. Una de las ultimas versiones que ha lanzado, es la que Vd. tiene en su Computador (version 2.2), con lo cual, sera con la que trabajaremos en este libro.

2.2 EL USO DEL DISCO DE CP/M

Lo primero que le recomendamos que Vd. debe hacer, es preparar un nuevo Disco en donde pueda escribir y leer sus propios programas, y para ello necesitara formatear el nuevo Disco bajo el Sistema Operativo de CP/M. Para realizar este formateado del nuevo Disco, debera cargar el Sistema Operativo CP/M, introduciendo el Disco que lo contiene en la unidad A y tecleando el comando que reclama al CP/M.

```
|cpm
```

Este comando se encarga de transferir el control que tenia el AMSDOS, para que lo asuma el CP/M. Transcurridos unos segundos, aparecera en pantalla el siguiente mensaje:

```
CP/M 2.2 - Amstrad Consumer Electronics plc.
A>
```

Con este mensaje, su Computador tiene el control directo del Sistema Operativo CP/M, con lo que una vez aparezca el signo A>, indicara que el sistema esta listo para recibir ordenes. Naturalmente, bajo este sistema no podra utilizar los comandos de Basic, dado que el CP/M no los interpretara.

Para poder conocer algunos de los comandos que si interpretara, asi como programas contenidos en el Disco, entre lo siguiente:

```
dir
```

En su pantalla aparecera el directorio del Disco de CP/M con todo su contenido, observando que en algunos programas, los nombres van seguidos con el sufijo COM, que corresponden a comandos directamente ejecutables, encontrandose entre ellos el comando "FORMAT", asi pues teclee lo siguiente:

```
format
```

Apareciendo en pantalla, el siguiente mensaje:

```
Please insert disc to be formatted into drive A
then press any key
```

Antes de presionar cualquier tecla, saque el disco de CP/M de la unidad A, e inserte el nuevo Disco en esa unidad, pudiendo en ese momento pulsar cualquier tecla, empezando la operacion del

formateo del Disco en ese mismo momento desde la pista 0 y terminando en la pista 39, con lo que su Computador, una vez terminada esa operacion le preguntara:

```
Do you want to format another disc (Y/N):
```

En este momento, si Vd. desea formatear otro Disco, debera sustituir el Disco que ha formateado por otro nuevo, y contestar al Computador con la letra Y (mayuscula). En el caso de que no quiera formatear ningun otro Disco, debera contestar al Computador con la letra N (mayuscula) con lo que el sistema le pedira con el siguiente mensaje, que inserte en la unidad A el Disco del Sistema Operativo CP/M:

```
Please insert a CP/M system disc into drive A
then press any key:
```

Una vez introducido el Disco con el CP/M, y habiendo presionado cualquier tecla, le aparecera en pantalla la informacion de que Vd. tiene control del sistema bajo el Sistema Operativo de CP/M.

2.3 COPIA DE DISCOS BAJO SISTEMA CP/M

Aunque mas adelante le indicaremos com copiar discos y ficheros, en esta seccion, le mostraremos un modo rapido y eficaz para la duplicacion de Discos.

Asumiendo que Vd. tiene control de CP/M en su Computador, usaremos una de las utilidades existentes en este sistema llamada DISCCOPY, que le permitira la copia de un Disco, por lo que debera introducir el siguiente comando una vez tenga en pantalla el signo A>, que le indica que puede mandar ordenes al Computador, asi pues teclee lo siguiente:

```
disccopy
```

El sistema le indicara que introduzca el disco original en la unidad A y luego pulse cualquier tecla, con lo que quite el disco de CP/M he inserte el que desea copiar antes de pulsar cualquier tecla. Si lo que Vd. quiere es copiar el mismo disco de CP/M, deje ese mismo disco como original y presione cualquier tecla, con lo que el Computador empezara su mision de copia, apareciendo en pantalla el siguiente mensaje:

```
Copying started
Reading track 0 to 7
```

queriendo decir que la operacion ha empezado leyendo las pistas del 0 al 7, del disco original, con lo cual las acumula en memoria para una vez terminada su lectura y almacenamiento, pedirle que inserte el disco de destino (disco formateado como copia del original) en la unidad A, y luego pulsar cualquier tecla, pero antes debera haber quitado de esa unidad el disco original, apareciendo en pantalla el siguiente mensaje:

```
Writing track 0 to 7
```

y asi sucesivamente hasta completar el disco, que debera terminar el la pista 39 como maximo, en saltos de 8 pistas. En ese mismo momento, le pidira el Computador, si desea copiar otro disco, con lo que Vd. respondera por el teclado con una Y (mayuscula = si) si desea continuar con otra copia, o con una N (mayucula = no) si desea terminar y no hacer mas copias. Esta utilidad, es usada solamente cuando se tiene instalada solamente una unidad de Disco, por lo que es una operacion un poco lenta.

En el caso de que Vd. quiera comprobar la copia que ha efectuado, debera usar la utilidad que tiene en su disco de CP/M llamada DISCCHK, con lo cual debera teclear lo siguiente, entendiendo que sigue con el control de CP/M:

```
discchk
```

Siga las instrucciones que le aparecen en pantalla, en relacion que disco debe introducir en la unidad, y si el Computador detecta alguna anomalia entre las pista que a leido del disco original y las del disco de origen, le aparecera un mensaje en pantalla que dira lo siguiente:

```
Failed to verify destination disc correctly:
track x sector y
```

Este mensaje significa que Vd. ha tenido un fallo en la verificacion del disco de destino, senalada como "pista x sector y", detectandose una diferencia de informacion entre los dos discos.

En el caso de que su Computador tenga conectadas dos unidades de Disco, Vd. podra usar la utilidad de COPYDISC que esta en su disco de CP/M para hacer las copias de los discos que desee, asi como una copia de trabajo de su mismo disco de CP/M. Para ello debera teclear lo siguiente, una vez tenga el control indicado por el signo de A> en el Sistema CP/M:

```
copydisc
```

Siga las instrucciones que le aparecen en la pantalla, y el contenido de su disco de origen, sera transferido al disco de destino, normalmente el la unidad B, transfiriendose 8 pistas en cada pasada, hasta que haya terminado con la ultima, que normalmente es la 39. Al igual que el DISCCOPY, tambien el COPYDISC incorpora el formateado automatico del disco, si no ha sido formateado anteriormente.

Al igual que con una unidad de disco, tenemos una utilidad de verificacion para dos unidades de disco, la cual se llama CHKDISC, pudiendose usar de la misma forma que la DISCCHK. Asi pues teclee la siguiente palabra:

```
chkdisc
```

siguiendo las intrucciones que aparecen en pantalla, para la introduccion de los discos en sus respectivas unidades. Los mensajes de error de verificacion, son los mismos que con la otra utilidad de verificacion DISCCHK.

En el caso de que Vd. quiera abandonar cualquiere de estas operaciones, lo unico que debera hacer sera: pulsar simultaneamente la tecla de CTRL y la tecla C, permitiendole regresar al modo directo de control de CP/M.



3.1 ORDENES PERMANENTES

Una de las operaciones, durante el proceso de la carga del Sistema Operativo CP/M en memoria, es la ubicacion de las ordenes permanentes, las cuales se ejecutan inmediatamente sin consultar ninguna instruccion de Disco.

Las ordenes permanentes mas comunes en este sistema son: DIR, TYPE, ERA, REN, y b:. Mas adelante, describiremos en este libro cada una de estas ordenes en detalle, pero como Vd. puede ver, solo necesita aprender unas pocas ordenes permanentes.

3.2 INTRODUCCION A LAS ORDENES

Antes de informarle lo que hace una orden, Vd. debe conocer los diversos convenios que usamos para las ordenes y las teclas, siendo estos los siguientes:

<Oprimir> Significa el presionar una sola tecla.
<Teclear> Significa el presionar una serie de teclas.
<cr> Este simbolo, significa el oprimir la tecla de RETURN.
[ctrl] X Significa el introducir un caracter de Control X.

Un ejemplo tipico de la introduccion de una orden, seria:

DIR <cr>

Significando que Vd. debera pulsar las teclas D, I y R, y luego presionar la tecla de RETURN.

Los caracteres de control, estan sujetos a la interpretacion especial del Computador. Al igual que Vd. mantiene presionada la tecla de SHIFT para poder imprimir las letras mayusculas, de la misma forma habra de oprimirse la tecla de control (CTRL) y la de la letra deseada para esta funcion.

3.3 EXPLICACION DE LAS ORDENES PERMANENTES

3.3.1COMANDO DIR

Este comando le listara todos los ficheros que se encuentren en el Directorio del Disco. Se permiten varias formas de ordenes

con este comando, para la exploracion de los Directorios de un disquete, usando caracteres ambiguos para relacionar un grupo de ficheros nombrados de forma similar. Estos ficheros no estan ordenados de ninguna manera particular, sino que la posicion de estos, indica unicamente la posicion que la resena de este fichero tiene en el Directorio. Este comando permite diferentes formas de expresarlo, las cuales son las siguientes:

DIR	Listara todos los ficheros resenados en la unidad de Disco principal (normalmente la unidad A).
DIR B:	Listara todos los ficheros resenados en la unidad de Disco B.
DIR *.BAS	Este comando, solamente le listara del Directorio, todos los ficheros de la clase .BAS de la unidad principal (tambien se pueden pedir los ficheros de otro tipo o clase como, .COM, .EXE, .ASM, etc).
DIR B:*.BAS	Este comando es identico al anterior, con la unica diferencia de que se requiere la informacion de la unidad de Disco B.
DIR PIP.COM	Listara solamente el fichero con el nombre PIP.COM si se encuentra en el Directorio de la unidad de Disco principal.

A continuacion, le mostraremos varios ejemplos del uso de este comando y sus resultados.

```
A> DIR B: <cr>
B: DOCUMENT.TXT
B: PRUEBA.ASM
B: PRUEBA.BAS
A>
```

Este ejemplo nos muestra el Directorio de la unidad de Disco B.

```
A> DIR B:*.BAS <cr>
B: FORMULA.BAS
B: INVENTA.BAS
B: COPIA.BAS
A>
```

En este ejemplo le mustran todos los ficheros de la clase .BAS, contenidos en el Directorio de la unidad de Disco B.

```
A> DIR PI??.* <cr>
A: PIP.COM
A>
```

Esta clase de ejemplo, la cual no ha sido explicada anteriormente, mostrara todos los ficheros que comiencen por PI (pudiendose usar cualquier otras dos letras) y de cualquier clase de programa, contenidos en la unidad de Disco principal.

3.3.2 COMANDO ERA

Este comando se usa para borrar, o eliminar, ficheros del Directorio, los cuales al no ser de su interes o utilidad, le dejaran espacio libre en el Disco. Se podra observar que este comando solo ejerce su funcion sobre el Directorio, de manera que la informacion del programa permanece en el Disco sin poder ser recuperada, pero pudiendo escribirse encima de ella, por lo que

resumiendo, este comando solo borra, o elimina los nombres de los ficheros del Directorio.

Es una buena costumbre que antes de borrar un fichero, o ficheros, use el comando DIR para verificar que el fichero a borrar se encuentra en ese disquete. Una vez hecha esta comprobacion, use el comando ERA para realizar el borrado, y por ultimo, use de nuevo el comando DIR para asegurarse de que el CP/M a borrado correctamente los ficheros elegidos.

Las diferentes formas de expresar este comando, son las siguientes:

ERA PIP.COM	Esta configuracion borrara el fichero PIP.COM del disquete de la unidad principal.
ERA B:PIP.COM	Esta forma de expresion es identica a la anterior, con la unica diferencia que se ejecuta en la unidad de Disco B.
ERA *.BAS	Borrara todos los ficheros de la clase .BAS del disquete de la unidad principal.
ERA B:*.BAS	Esta forma es identica a la anterior, con la unica diferencia que el borrado lo hace en la unidad de Disco B.

He aqui algunos ejemplos del uso de la orden ERA.

```
A> DIR B: <cr>
B: PIP.COM
B: PRUEBA.BAS
B: PRUEBA.ASM
A> ERA B:PRUEBA.BAS <cr>
A> DIR B: <cr>
B: PIP.COM
B: PRUEBA.ASM
A>
```

En este ejemplo, primero se usa el comando DIR para inspeccionar el Directorio de la unidad B, encontrando tres ficheros. El siguiente paso, es el borrar de la unidad B, el fichero PRUEBA.BAS, y finalmente se vuelve a usar el comando DIR, para comprobar si efectivamente se ha borrado.

```
A> DIR B:*.BAS <cr>
B: PRUEBA.BAS
B: EJEMPLO.BAS
A> ERA B:*.BAS <cr>
A> DIR B:*.BAS <cr>
Not Found
A>
```

En este ejemplo encontramos con el comando DIR, dos ficheros del tipo .BAS en la unidad de Disco B, pidiendole al CP/M que borre los ficheros de la clase .BAS en la unidad B. A continuacion, usando el comando DIR, le pedimos que inspeccione el Directorio para que nos confirme que no hay ningun fichero de la clase .BAS.

```
A> DIR <cr>
A: PIP.COM
A: PRUEBA.BAS
A: ASM.COM
A> ERA *.* <cr>
All files (Y/N)? Y <cr>
A> DIR <cr>
No file
A>
```

Este ejemplo demuestra lo que sucede cuando Vd. pide al CP/M que borre todos los ficheros de un disquete. Podemos notar el mensaje "All files (Y/N)?" y de la respuesta del usuario. En el caso de que Vd. responda con otra letra que no sea Y o N, no se borrara ningun fichero del disquete, siendo en este caso la respuesta de "Y" para que el comando ERA borre todos los ficheros.

3.3.3 COMANDO REN

Este tipo de comando, tiene la funcion de poder cambiar el nombre de un fichero ya esistente en Disco. Vd. debera mencionar el nombre antiguo del fichero y el nuevo nombre, no pudiendo usar los indicadores ambiguos * y ? para volver a dar nombre a un grupo de ficheros al mismo tiempo.

Cuando la Computadora descifra una sentencia de equivalencia, como el comando de renombrar, la nueva formacion va a la izquierda de un signo de igualdad (=) y la antigua aparecera a la derecha.

REN B:Nuevonombre.Tipo=Antiguonombre.Tipo <cr>

En este comando, se le puede especificar la unidad de Disco donde el fichero, o ficheros, se encuentran para ser renombrados. En el caso de no especificar la unidad de Disco, el sistema interpretara como que debe dirigirse a la unidad principal.

Observemos el siguiente ejemplo:

```
A> DIR <cr>
A: PIP.COM
A: ASM.COM
A: PRUEBA.BAS
A> REN NUEVO.BAS=PRUEBA.BAS <cr>
A> DIR <cr>
A: PIP.COM
A: ASM.COM
A: NUEVO.BAS
```

En este ejemplo, lo primero examinaremos él directorio del disquete para ver si se encuentra el fichero deseado, a continuacion, cambiaremos el nombre de dicho fichero (PRUEBA. BAS por NUEVO.BAS), inspeccionando de nuevo el directorio para ver si se ha efectuado el cambio de nombre, con el comando DIR.

3.3.4 COMANDO TYPE

Este comando, le permite visualizar un fichero que contiene informacion en codigo ASCII. Normalmente este comando se aplica

a ficheros de la clase "BAS", "ASM", "BAK", "DAT", "HEX", "DOC", "TXT", o cualquier otro tipo de fichero que contemga texto ASCII o datos.

En el caso de que el fichero que se quiere visualizar no es uno de la clase antes mencinada, le informaremos que le apareceran en pantalla efectos y letras indescifrables y de poco interes para Vd.

Solamente hay una forma del comando TYPE:

```
TYPE B:Nombredefichero.Tipo <cr>
```

Le visulizara el contenido de dicho fichero.

3.3.5 COMANDO b:

En el Sistema Operativo CP/M instalado en su Computadora con dos unidades de Disco, se puede cambiar la unidad activa o principal, escribiendo la letra que representa a la nueva unidad, seguida de dos puntos (:).

Para demostrar el uso de esta funcion, pondremos el siguiente ejemplo:

```
A> B: <cr>
B>
```

Para pasar a la unidad A, escribir:

```
B> A: <cr>
A>
```



4.1 ORDENES DE EDICION DE LINEA

Las ordenes de edicion de linea, tambien llamadas, ordenes, o codigos de control de consola, le permiten el controlar los errores mecanograficos durante la introduccion de linea de orden, dando control sobre el terminal de consola.

4.1.1 CONTROL-C [CTRL] C

El Control-C restaura la informacion interna del CP/M (arranque caliente), a un estado predefinido, sin destruir los programas, o datos almacenados en memoria. Esta orden, tiene dos usos principales:

1.- Catalogar un disquete cuando Vd. inserta otro disquete en una o mas unidades.
2.- Interrumpir el programa transitorio y volver al nivel de ordenes de CP/M.

El no cerrar un fichero en un momento adecuado, le puede crear problemas, pudiendo salvar estos, permitiendo siempre a un programa que termine normalmente (si en el programa hay una opcion QUIT, se podra usar esta, en lugar de presionar el RESET o Control-C).

4.1.2 CONTROL-E [CTRL] E

El Control-E, le permite continuar escribiendo en la siguiente linea, asi pues, el cursor se desplazara al comienzo de la linea siguiente.

El propio caracter Control-E, no se considera parte de la linea de orden resultante.

4.1.3 CONTROL-P [CTRL] P

Este control, le asignara la impresora para salida de informacion, con lo que toda salida que va a parar a la pantalla se enviara tambien a la impresora.

El Control-P actua como si fuese un conmutador que cambia de un estado "ON" a "OFF" (o de "OFF" a "ON" segun se encuentre en ese momento), cambiando de un estado a otro oprimiendo una vez el [CTRL] P (o sea para activar la impresora o desactivarla).

Existen varios inconvenientes al usar este tipo de Control, que son los siguientes:

1.- Su impresora no paginara, simplemente copiara una linea tras otra, sin tener en cuenta el fin de pagina.
2.- Cualquier codigo de control insertado en el texto, puede afectar a la impresora, haciendo cosas raras.
3.- La salida por pantalla se realizara, al menos que Vd. tenga una impresora rapida, esperando el Computador a enviar un caracter si encuentra ocupada la impresora.

Algunos programas desactivan automaticamente la impresora, se haya seleccionado o no, tales com los procesadores de texto que tiene sus propios controles de impresora. Nunca teclee el [CTRL] P durante la ejecucion de programas que tengan controles para el manejo de la impresora, dado que el resultado seria nefasto.

4.1.4 CONTROL-S [CTRL] S

Este tipo de control, es para la salida de informacion por la pantalla, dado que presionando una vez las teclas de CTRL y S la suspension de salida se activa, pudiendo reanudarla, presionando nuevamente cualquier tecla. La razon por la cual la pantalla se desactiva, es que a su vez la ejecucion de la rutina que en ese memento se estaba ejecutando, se ha parado, por lo que el ¡CTRL¿ S no actua sobre el hardware de la seccion de video, si no que afecta a la ejecucion del programa en memoria.

4.1.5 CONTROL-Z [CTRL] Z

Lo unico que hace el CONTROL-Z, es finalizar el texto que se estaba introduciendo en memoria.

4.2 LOS FICHEROS DE CP/M

Los ficheros de CP/M son identificados por un nombre de fichero y un tipo o clase de programa. Todos estos identificadores se detallan a continuacion.

4.2.1 NOMBRES DE LOS FICHEROS

En CP/M, cada fichero tiene un nombre unico con un maximo de ocho caracteres, pudiendose usar, tanto letras mayusculas, como minusculas. El CP/M convierte automaticamente los caracteres entrados por teclado, en una linea de orden, desde las letras mayusculas a las minusculas, no aceptando espacios en blanco ni caracteres como:

< > . , ; : = ? * [ ] ( ) / o <TAB>

Tampoco se deberan usar caracteres de CONTROL en el nombre de un fichero (o caracteres que no aparezcan en pantalla). Una recomendacion para elejir un nombre para un fichero, es que dicho nombre este relacionado con el tipo de programa, como por ejemplo: un programa de contabilidad, se le podria llamar CONTAB.

4.2.2 CLASES DE FICHEROS

Es norma en CP/M, el identificar los ficheros para que de esa manera saber de que clase o tipo de ficheros se tratan y para que deben ser usados, ya que tipos diferentes de informacion, se manejan de diferente forma y con procedimientos distintos, como en el caso de que ciertas operaciones sobre ficheros programa pueden destruir los contenidos de un fichero de datos, si por error son accedidos.

Para que no ocurran estos desastres, el CP/M cuenta con el tipo de fichero para expresar la funcion del mismo, siguiendo al nombre del fichero, con un punto delante, constando de tres caracteres como maximo, identificando el sistema al fichero por el punto y el tipo o clase de programa asi como por su nombre.

A continuacion, le detallamos los diferentes tipos de ficheros usados comunmente por los usuarios de este tipo de sistema CP/M:

.ASC	Fichero que contiene texto en ASCII.
.ASM	Fichero de un programa fuente en leguaje ensamblador.
.BAK	Fichero copia
.BAS	Fichero de un programa fuente en Basic.
.CAL	Fichero de datos Supercalc
.COM	Programa transitorio directamente ejecutable
.DAT	Fichero de datos.
.DOC	Fichero de documentos.
.HEX	Fichero de codigo objeto en formato hexadecimal.
.INT	Programa en codigo intermedio Basic (CBASIC).
.IRL	Fichero de libreria indexada.
.LIB	Fichero de libreria.
.MAC	Programa fuente en lenguaje macroensamblador.
.OBJ	Programa en codigo maquina.
.OVR	Fichero de sobrecarga.
.PAS	Programa fuente en Pascal.
.PRL	Fichero reubicable por paginas.
.PRN	Fichero de listado en lenguaje ensamblador.
.REL	Fichero de codigo maquina reubicable.
.SRC	Fichero fuente.
.SUB	Fichero de ordenes para un programa SUBMIT.
.SYS	Fichero del sistema.
.TEX	Fichero de un documento
.TXT	Fichero de un documento (normalmente texto).
.$$$	Fichero temporal no usable.

Ademas de todos estos tipos de ficheros, Vd. puede crear su propio tipo para un programa especifico o diferente version del mismo. Por ejemplo, si se esta trabajando sobre un programa en Basic llamado PRUEBA, la lista de los ficheros de este programa podria ser la siguienteL:

```
PRUEBA.001
PRUEBA.002
PRUEBA.003
PRUEBA.BAS
```

Representando el fichero PRUBA.BAS, el programa terminado y pulido para ser ejecutado.

CAPITULO 5

5.1 ORDENES TRANSITORIAS

Como recordaremos, las ordenes permanentes son aquellas palabras que el CP/M puede interpretar usando solamente las instrucciones ya presentes en memoria, mientras que las ordenes transitorias (normalmente llamadas programas) se ejecutan, escribiendo el nombre, el cual actua como una orden para que el CP/M obtenga informacion adicional del disquete.

Vd. podra comprobar las ventajas de las ordenes transitorias del CP/M, dado que si el fichero que coincide con su orden contiene instrucciones para el Computador, parecera exactamente igual que si se hubiese dado una orden valida en CP/M, aunque podra tardar un poco mas tiempo su ejecucion, dado que tiene que acceder al disquete para obtener la informacion, y poder empezar a trabajar.

A partir de ahora definiremos "Programa" en el contexto de CP/M de la forma siguiente: un programa es un conjunto de instrucciones almacenadas en un disquete en ficheros del tipo ".COM", pidiendolo solamente por su nombre (o sea sin usar el tipo .COM), denominando a este tipo, programas transitorios u ordenes transitorias. Significando que una vez que un programa se carga del disquete a la memoria, esta, temporalmente, ignora el CP/M y obedece las instrucciones del programa.

5.2 UTILIDADES DE MANTENIMIENTO

Las utilidades de Mantenimiento, sirven para mantener el buen estado de nuestros Discos, debiendo realizar las siguientes tareas:

Averiguar el estado actual del Disco.
Reorganizar la informacion del Disco.
Expandir las capacidades de archivo.

Dado que las ordenes permanentes realizan estas tareas, podriamos categorizar dichas ordenes por funciones de la siguiente forma:

DIR	=	Estado
ERA	=	Reorganizacion
REN	=	Reorganizacion
SAVE	=	Reorganizacion y Expansion
TYPE	=	Estado

Sin embargo, estas ordenes no son suficientes para alterar el disco o para obtener mejor reordenacion, dado que si por ejemplo, mientras se puede ver que ficheros estan en un disquete, estas ordenes no pueden ver, ni determinar el tamano de los mismos.

Para esto se han creado las ordenes de Mantenimiento, las cuales son una serie de utilidades que ayudan a mantener sus disquetes en orden. Estas utilidades son las siguientes y pueden categorizarse por funciones:

SYS FICHERO DEL SISTEMA.
Los ficheros SYS nunca apareceran cuando se le pide al sistema un Directorio (DIR). En este tipo de fichero, se puede leer y escribir informacion, asi como borrarlo, o usarlo de otra manera normal sin aparecer nunca en el Directorio. Sin embargo se podra visualizar cuando se usa el STAT para revisar el estado de los ficheros.

DIR FICHERO DE DIRECTORIO.
El fichero DIR aparecera siempre que se trate de acceder al Directorio del area de usuario, siendo el atributo normal dado a un fichero.

5.3.1 EL USO DE STAT

STAT <cr>

Indica el espacio libre que hay en los discos desde el ultimo arranque en frio o caliente.

Ejemplo:

```
A> STAT <cr>
A: R/W, SPACE: 2K
A>
```

```
A> STAT <cr>
A: R/W, SPACE: 2K
A: R/O, SPACE: 120K
A>
```

STAT d: <cr>

Indica el espacio libre disponible en el Disco del dispositivo d:

Ejemplo:

```
A> STAT B: <cr>
BYTES REMAINING ON
B: 170K
A>
```

STAT d:fichero.tipo <cr>

Este comando le indicara el espacio que ocupa el fichero "fichero.tipo" en el dispositivo d:. Tanto el fichero como su extension podran contener caracteres ambiguos (* y ?) del CP/M. La indetificacion del dispositivo sera opcional, y si se omite, el sistema entendera que es la unidad principal que contiene el control.

Ejemplos:

A> STAT JUAN.TXT <cr>

```
RECS   BYTS   EX   ACC    D:FILENAME.TYP
 5      2K    1    R/W    A:JUAN.TXT
A>
```

STAT	=	Estado, Reorganizacion, y Expansion
PIP	=	Reorganizacion, y Expansion
ED	=	Reorganizacion, y Expansion
DUMP	=	Estado
SYSGEN	=	Reorganizacion, y Expansion
MOVCPM	=	Reorganizacion, y Expansion

5.3 STAT - ESTADISTICAS DE FICHEROS

La palabra STAT, significa ESTADISTICA, suministrando informacion mas completa sobre un fichero o un grupo de ficheros de un Directorio, asi como la gestion de perifericos.

Las ordenes de STAT suministran, o bien sobre el tamano y atributos de un fichero o ficheros, o ademas pueden cambiar sus atributos. Los diferentes parametros de un fichero son los siguientes:

TAMANO DE UN FICHERO (File size): Se refire a la cantidad de espacio (octetos) que ocupa un fichero.

ESPACIO LIBRE (Free space): Significa la cantidad de espacio disponible (en octetos) en el Disco, para poder almacenar ficheros de informacion.

ATRIBUTOS DE DISCO O FICHERO: Estos atributos son un conjunto especifico de caracteristicas que podemos asignar a un fichero o disquete.

El tamano de un fichero, y la cantidad de espacio libre en un disquete es indicado por el STAT en octetos, siendo este una unidad de memoria capaz de almacenar un solo caracter, lo que significa que 1K octeto es igual a 1.024 octetos. Si en la informacion que nos da el STAT nos indica que tenemos de espacio libre 34K, esto quiere decir que podemos almacenar informacion en el disquete 34.816 (1.024 por 34) octetos.

Los atributos que se pueden asignar en el STAT con el CP/M a un fichero son: R/O o R/W y DIR o SYS.

R/O FICHERO DE LECTURA SOLAMENTE.
Este tipo de fichero, no se podra actualizar o borrar con la orden ERA, asi mismo tampoco se podra escribir o introducir informacion en este tipo de fichero, protegiendolo contra borrados accidentales.

R/W FICHERO DE LECTURA Y ESCRITURA.
Este tipo de fichero es lo opuesto al anterior, pudiendo actualizarlo o borrarlo, dado que no esta protegido para la escritura, a menos que el disquete tenga su proteccion colocada para no por acceder a el en escritura.

```
A> STAT B:JUAN.TXT <cr>
```

```
RECS   BYTS   EX   ACC    D:FILENAME.TYP
 10     4K    1    R/W    B:JUAN.TXT
A>
```

```
A> STAT *.COM <cr>
```

```
RECS   BYTS   EX   ACC    D:FILENAME.TYP
  4     2K    1    R/O    A:PIP.COM
  3     2K    1    R/W    A:DUMP.COM
 50     6K    1    R/W    A:ED.COM
 30     4K    1    R/W    A:STAT.COM
BYTES REMAINING ON A:  210K
A>
```

```
A> STAT B:PRUEBA.?X? <cr>
```

```
RECS   BYTS   EX   ACC    D:FILENAME.TYP
 12     3K    1    R/W    B:PRUEBA.TXT
 24     4K    1    R/W    B:PRUEBA.EXT
A>
```

STAT d:fichero.tip $atr <cr>

Esta orden asigna el atributo "atr" (R/O, R/W, DIR o SYS precedido por el signo de dolar) al fichero o ficheros en el dispositivo d:, pudiendose utilizar los especificadores ambiguos (* y ?). En el caso de que no se especifique el dispositivo d:, el sistema interpretara que es la unidad principal activada.

Ejemplos:

```
A> STAT PRUEBA.TXT $R/O <cr>
```

```
PRUEBA.TXT SET TO R/O
A>
```

```
A> STAT B:PRUEBA.TXT $SYS <cr>
```

```
B:PRUEBA.TXT SET TO SYS
A>
```

```
A> STAT *.* $DIR <cr>
```

```
PRUEBA.TXT SET TO DIR
PRUEBA.EXT SET TO DIR
A>
```

5.3.2 STAT - DISPOSITIVOS

La funcion STAT, tambien suministra informacion sobre los dispositivos logicos y fisicos del sistema. Un dispositivo logico es una funcion general de su Computadora, mientras que un dispositivo fisico es una parte concreta del equipo que se define para hacer una funcion.

Para indicar al sistema nuestra eleccion, usaremos la orden STAT (en dispositivos), habiendo cuatro dispositivos logicos disponibles:

```
CON: Mete ordenes y visualiza la informacion.
RDR: Recibe la informacion.
PUN: Manda la informacion.
LST: Lista o imprime la informacion.
```

Los dispositivos fisicos son los siguientes:

```
TTY: Consola de visualizacion (teleimpresora)
CRT: Consola de visualizacion (Pantalla)
BAT: Procesador de lotes
LPT: Impresora
UL1: Dispositivo de listar
```

STAT DEV: <cr>

Esta orden hace visible la asignacion actual de los dispositivos.

Ejemplo:

```
A> STAT DEV: <cr>
CON: IS CRT
RDR: IS UL1
PUN: IS BAT
LST: IS LPT
A>
```

STAT VAL: <cr>

Este tipo de orden, visualiza las posibles asignaciones de los dispositivos fisicos y logicos, ademas de un resumen de las ordenes abreviadas de STAT.

Ejemplo:

```
A> STAT VAL: <cr>
TEMP R/O DISK: D:=R/O
SET INDICATOR: D:FILENAME.TYP $R/O $R/W $SYS $DIR
DISK STATUS :DSK: D:DSK:
USER STATUS :USR:
IOBYTE ASSIGN:
CON:= TTY:CRT:BAT:UC1:
RDR:= TTY:PTR:UR1:UR2:
PUN:= TTY:PTP:UP1:UP2:
LST:= TTY:CRT:LPT:UL1:
```

Nota:
Las primeras lineas indican las posibles ordenes de los dispositivos STAT:

STAT D:= R/O	Recepcion temporal de Disco
STAT D:FILENAME.TYP R $ATR	Colocar atributo
STAT D:DSK:	Ver estadisticas de un Disco
STAT D:USR:	Ver estadisticas del area del usuario

STAT log:= phy: <cr>

Asigna el dispositivo fisico al logico especificado.

Ejemplo:

```
A> STAT CON:=CRT:,LST:=UL1: <cr>
A>
```

Hay que anotar, que se pueden hacer varias asignaciones de diferentes dispositivos, separados cada uno de ellos por una coma.

STAT USR: <cr>

Este comando le saca por pantalla el numero del actual usuario y le lista los ficheros que tienen el numero de usuario.

Ejemplo:

```
A> STAT USR: <cr>
ACTIVE USER: 0
ACTIVE FILES: 0 1 5
A>
```

Este mensaje, le indica que esta en el area de usuario 0, pero que hay ficheros en las areas 0, 1 y 5.

STAT d:DSK: <cr>

Esta orden le indica el como se almacenan los datos en el Disco del dispositivo d:.

Ejemplo:

```
A> STAT B:DSK: <cr>
B: DRIVE CHARACTERISTICS
    4096: 128 BYTE RECORD CAPACITY
     512: KILOBYTE DRIVE CAPACITY
     128: 32 BYTE DIRECTORY ENTRIES
     128: CHECKED DIRECTORY ENTRIES
     128: RECORDS/EXTENT
      16: RECORDS/BLOCK
      58: SECTORS/TRACK
       2: RESERVED TRACKS
```

Nota:
La esplicacion de cada uno de los apartados anteriormente descritos, significa lo siguiente:

128 BYTE RECORD CAPACITY: Es el maximo numero de registros de 128 octetos que se pueden almacenar en un disco.

KILOBYTE DRIVE CAPACITY: Es el maximo numero de K octetos que se pueden almacenar en un disco.

32 BYTE DIRECTORY ENTRIES: Numero maximo de ficheros que se pueden meter en un disco.

CHECKED DIRECTORY ENTRIES: Su significado es lo mismo que el anterior.

RECORDS/EXTENT: Maximo numero de registros por entrada.

RECORDS/BLOCK: Espacio maximo de disco que se puede permitir en un fichero.

SECTORS/TRACK: Numero de sectores en que se puede dividir una pista.

RESERVED TRACKS: Numero de pistas reservadas de disco que no estan disponibles para el almacenamiento de ficheros.

STAT d: = R/O <cr>

Esta orden asigna temporalmente una proteccion de escritura al disco del dispositivo d:.

Ejemplo:

```
A> STAT B:=R/O <cr>
A> SAVE 2 B:PRUEBA.FIL
BDOS ERR ON B: R/O
```

En este ejemplo, lo primero que hacemos es actualizar el dispositivo B:, para poder leer solamente, sin darnos confirmacion de nuestra peticion. Seguidamente guardamos el nuevo fichero en disco y recibir un mensaje de BDOS ERR el cual nos indica que el CP/M ha considerado el disco protegido contra escritura.

CAPITULO 6

6.1 PIP - COPIADOR DE INFORMACION

Esta orden transitoria sirve para copiar informacion de un sitio a otro, y sus siglas representan, Peripheral Interchange Program. Los nombres de los ficheros que se desean copiar, pueden ser ambiguos, pudiendose utilizar parametros opcionales. Asi mismo, tambien sirve para transferir informacion entre la unidad central y sus perifericos, siendo la forma general de esta orden: PIP <destino>=<origen>.

Hay dos formas de poder llamar a este programa, y son las siguientes:

```
A>PIP pipcomando <cr>
```

Si se desea realizar varias operaciones, se puede llamar a este programa con esta segunda opcion:

```
A>PIP <cr>                      Orden de CP/M
*pipcomando <cr>                Orden de PIP
.
.                               Diferentes ordenes de PIP
.
*pipcomando <cr>                Orden de PIP
*[C]                            Control C hace retornar a CP/M
A>
```

Las ordenes y parametros especiales de PIP, indican la transferencia realizada entre dos ficheros o grupo de ficheros, en cada caso el fichero original permanece intacto, pudiendose duplicar un disquete entero o una porcion de un fichero.

Los diferentes usos del PIP, se detallan a continuacion:

1.- Hacer visible los contenidos del fichero copiado.
2.- Copiar un fichero de un disco a otro.
3.- Copiar un fichero R/O
4.- Crear un fichero con un nombre diferente sin variar su contenido
5.- Copiar un sistema de ficheros.
6.- Copiar varios ficheros de un disco a otro.
7.- Copiar una parte de un fichero.
8.- Copiar todos los ficheros de un disco a otro.
9.- Crear un fichero mediante una concatenacion de otros varios.

Ademas de lo anteriormente mencionado, la orden PIP, puede copiar datos de un fichero a un dispositivo y viceversa, asi como entre dispositivos, sutituyendo el nombre del fichero por el dispositivo tanto en la fuente como en el destino, o entre ambos.

A continuacion se detallan algunos usos del PIP como dispositivo:

1.- Se puede enviar el contenido de un fichero a un dispositivo.
2.- Transferir datos de un dispositivo a un fichero.
3.- Transferir datos entre dos dispositivos.
4.- Imprimir un fichero con el formato de impresora o tamano de papel.
5.- Hacer salir por pantalla los datos que entran en la entrada del dispositivo.
6.- Cambiar las letras mayusculas en minusculas y viceversa.
7.- Poder almacenar en un fichero, todos los datos de entrada a un dispositivo.

Ademas de lo anteriormente mencionado, la orden PIP tiene cinco variantes especiales para los dispositivos, los cuales son, ademas de los mencionados en la seccion STAT:

NUL: Es un dispositivo que manda 40 caracteres nulos al destino indicado.
EOF: Dispositivo que manda una marca de fin de fichero al destino indicado.
OUT: Este dispositivo de destino esta hecho para el usuario, y el PIP debera ser modificado para que se incluya este.
INP: Este dispositivo fuente esta hecho para el usuario, y el PIP debera ser modificado para que se incluya este dispositivo.
PRN: Esta es una forma especial, parecida al LST, la cual expande las tabulaciones, numero de lineas y formatea la copia.

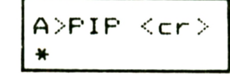
6.2 USO DE LA ORDEN PIP

A continuacion examinaremos los diferentes usos que tiene la orden PIP usando los diferentes parametros y opciones antes mencionados.

PIP <cr>

Esta orden, carga el programa PIP en memoria.

Ejemplo:



```
A>PIP <cr>
*
```

Cuando aparece el signo *, significa que el PIP esta listo para recibir ordenes validas. En el caso de querer volver al CP/M, habra que pulsar Control C.

PIP d:nuevo.tip=d:viejo.tip[p] <cr>

Esta orden copia el fichero viejo.tip de la unidad especificada en el fichero nuevo.tip en la unidad especificada, con los parametros [p] (estos parametros se detallaran mas adelante).

Ejemplos:

```
B>PIP CARTA2.DOC=CARTA1.DOC <cr>
B>
```

Este ejemplo es un simple uso de esta orden, la cual copia un fichero en un nuevo fichero con un nuevo nombre, en el disco B.

```
B>PIP A:CARTA2.DOC=B:CARTA1.DOC <cr>
B>
```

Esta orden, copia entre los discos A: y B: el fichero con un nuevo nombre.

```
B>PIP CARTA2.DOC=CARTA1.DOC[V] <cr>
B>
```

Este ejemplo es identico al primero, con la unica variacion que se incluye el parametro [V] para que verifique la copia.

```
A>PIP B:=A:CARTA.LET[V] <cr>
A>

A>PIP B:CARTA.LET=A:[V] <cr>
A>
```

Estos dos ejemplos hacen la misma funcion, siendo metodos abreviados para especificar el nombre de un fichero que se va a utilizar en las dos unidades de Disco (A: y B:).

```
A>PIP B:=A:*.*[V] <cr>
COPYING
CARTA1.DOC
CARTA2.DOC
CARTA1.LET
A>
```

Este metodo simplificado de copiar el disco A: en el disco B:, es muy comun, dado que informa en pantalla de lo que esta copiando.

PIP d:nuevo.tip=d:viejo1.fic[p],d:viejo2.fic[p] <cr>

Este tipo de orden, crea un fichero en la unidad especificada, constando este de dos ficheros distintos, separados por una coma.

Ejemplo:

```
A>PIP B:NUEVO=A:CARTA1[V],A:CARTA2 <cr>
A>
```

En este ejemplo se combinan los datos de los ficheros CARTA1 y CARTA2, en un nuevo fichero llamado NUEVO.

PIP dis:=d:fichero.tip[p] <cr>

Este metodo, envia el contenido del fichero de la unidad especificada, a otro dispositivo.

Ejemplo:

```
B>PIP LST:=B:CARTA.DOC <cr>
B>
```

En este ejemplo, el contenido del fichero CARTA.DOC es enviado al dispositivo LST.

6.3 LOS PARAMETROS DEL PIP

En la orden PIP, esisten diferentes parametros opcionales, los cuales se pueden introducir en la orden de PIP tanto por separado como agrupados, metidos entre corchetes ¡PV¿. No se asuste de la cantidad de parametros, dado que probablemente Vd. solo usara un par de ellos.

PARAMETRO [B]

La funcion de este parametro, es la transferencia de bloque, el cual transfiere la informacion de la fuente a un buffer hasta que reciba el caracter 53 hex.

La utilizacion de este formato es usada en aquellos dispositivos fuente que transfieren datos continuamente. Para operaciones normales de fichero a fichero no es necesario su uso.

PARAMETRO [D#]

Este parametro suprime todos los caracteres que sobrepasen la columna #-esima, indicando al PIP que suprima cualquier caracter que sobrepase una columna determinada. El numero de la columna debera seguir a la letra D, como por ejemplo [D18], siendo su mision orientada a trabajos de datos a linea. El valor de # debera ser un entero que este comprendido entre 1 y 255 ambos inclusive.

El uso de este parametro se utiliza principalmente para enviar la salida de una linea ancha a un dispositivo que maneje lineas estrechas como por ejemplo: una impresora, utilizandose para transferir fichero a fichero.

PARAMETRO [E]

Este parametro te da un eco en la pantalla al realizar una copia enviada a un dispositivo de destino. Este es un metodo muy practico, para revisar la informacion que esta saliendo.

PARAMETRO [F]

Esta opcion, filtra los caracteres de alimentacion de forma (OC hex) del flujo de datos, ignorando el PIP los caracteres ASCII definidos y hace la copia sin ellos. Estos alimentadores de forma, se tienen a menudo en ficheros para formatear la salida de la impresora.

El caracter de alimentacion de forma, controla la distribucion de la informacion canalizada visualmente o por impresora.

PARAMETRO [G#]

La mision de este parametro, es la de permitir la copia de ficheros de otras areas a la actual. El numero que hay que

introducir, ira desde el 0 al 15 ambos inclusive, representando este numero el area fuente del usuario. Este numero debera ir a continuacion de la letra G, como por ejemplo: G10.

PARAMETRO [H]

La funcion de este parametro, es la de transferir los datos en formato hexadecimal en vez de formato binario o ASCII.

Cuando en este formato, se detectan errores en hexadecimal, aparecera una peticion de orden para la accion correctiva a tomar.

PARAMETRO [I]

Este parametro se aplica a los registros en formato hexadecimal, indicando al programa PIP que ignore cualquier dato que aparezca en un registro nulo.

En el caso que se use la opcion [I], PIP pondra automaticamente la opcion [H].

PARAMETRO [L]

La funcion de este parametro, es la de poder convertir las letras de la A a la Z de mayusculas a minusculas durante el proceso de la copia.

Se debera tener mucha precaucion al utilizar esta opcion, si en el caso de que los ficheros a copiar contengan instrucciones de la Computadora.

PARAMETRO [N]

La mision de este parametro, es la de ir sumando el numero de lineas que se estan transfiriendo cada vez que el sistema detecta un retorno de carro, incrementando el contador de linea. Esta opcion es de gran uso, cuando se esta transfiriendo informacion a una impresora.

PARAMETRO [O]

El uso de esta opcion, esta destinado a la transferencia de ficheros en codigo objeto o de otros ficheros no ASCII, asi como de dispositivos que no envien datos ASCII.

Cuando se estan copiando ficheros del tipo .COM, no sera necesario el usar este parametro, dado que el PIP reconoce que estos ficheros no estan en ASCII.

PARAMETRO [P#]

Este parametro, indica al PIP que se quiere poner una alimentacion de forma, despues de una serie de lineas de datos, para eso se habra de poner una numeracion al lado de la P, que podra ser del 1 al 125 ambos inclusive, como por ejemplo: [P34].

Si se usan conjuntamente los parametros [F] y [P] se anulara la paginacion que implica la presencia de los caracteres de alimentacion de forma en la fuente de datos.

PARAMETRO [R]

Esta opcion, permite que solamente anadiendo una linea con esta opcion, podremos copiar ficheros, copiando el atributo siempre que este presente.

PARAMETRO [U]

Este parametro es el opuesto al [L], y convertira los caracteres con letras minusculas a mayusculas de la A a la Z durante el proceso de la copia.

PARAMETRO [V]

El uso de este parametro es muy util, dado que verifica que la copia es correcta, comparando el buffer de memoria con el fichero o ficheros creados.

PARAMETRO [W]

La funcion de este parametro, tambien es muy util, dado que le permite el copiar un fichero con el atributo R/O, pues esta opcion permite escribir en un fichero con R/O.

CAPITULO 7

7.1 ED - EDITOR DE FICHEROS DE TEXTO

El programa ED, se encuentra en el directorio bajo el nombre de ED.COM, pudiendose cargar directamente en memoria. Este programa consta de una serie de ordenes para poder editar ficheros de texto.

El Editor es un programa que transfiere los caracteres introducidos por el teclado, y enviandolos a un fichero en disco, asi pues, como se puede cometer algunos errores al teclear el texto, este programa contiene una serie de comandos, los cuales le permiten hacer todas las correcciones necesarias, asi como, suprimir o anadir informacion a ese fichero.

Para empezar a usar el Editor, habra que introducir primeramente el nombre del fichero (a esto se le llama, crear un fichero), por lo que teclearemos ;o siguiente:

```
A> ED nombre.tip <cr>
```

Una vez cargado el editor y creado el fichero, se podra comenzar a usar dicho programa, asi como el usar las ordenes permanentes del Editor para:

1.- Suprimir o cambiar cualquier parte del fichero creado.
2.- Insertar informacion nueva desde el teclado o desde otro fichero.

Supongamos que queremos editar en el fichero PROGRAMA.DOC, para entrar en el Editor, deberemos teclear lo siguiente:

```
A> ED PROGRAMA.DOC <cr>
```

ED (Editor) creara un fichero temporal llamado PROGRAMA.$$$, el cual no contiene informacion, si no solamente un directorio de entrada. Cuando se esta introduciendo el texto se transferira del fichero original al buffer del editor, el cual tiene una capacidad limitada, por lo que una vez lleno, se llevara todo la informacion al fichero temporal (PROGRAMA.$$$).

En el momento de que Vd. termine de Editar, parte del texto permanece en el fichero original (PROGRAMA.DOC), otra parte en el buffer del Editor, y la otra en el fichero temporal (PROGRAMA.$$$). La reconstruccion de estas tres partes, se efctua de la siguiente forma: Primero el ED toma el contenido del buffer, y lo lleva al fichero temporal (PROGRAMA.$$$), llevando el resto del texto del fichero original al fichero temporal, Finalmente, el fichero PROGRAMA.DOC (original), cambia su nombre por el de PROGRAMA.BAK y el fichero PROGRAMA.$$$ se le denominara PROGRAMA.DAY. Resumiendo, Vd. tendra una copia de la entrada de texto original en el fichero PROGRAMA.BAK, y la ultima version editada estara en el fichero denominado PROGRAMA.DAY.

7.1.1 ORDENES DE ED.COM

Para la utilizacion de las ordenes de ED.COM, se debera pedir al CP/M lo siguiente:

```
A>ED d:nombre.tip <cr>
*comandoED <cr>
```

De lo que deduciremos que, ED es el nombre del programa (ED.COM), d: es la unidad de Disco, que en el caso de no especificarla, el sistema se dirigira a la unidad de control, y por ultimo nombre.tip que es el fichero que hemos creado. Una vez presionado el <cr>, aparecera el signo *, indicandonos que el Editor esta listo para recibir ordenes, por lo que teclearemos la orden necesaria, seguida de un <cr>.

En esta seccion, se utilizaran diferentes abreviaturas, las cuales son las siguientes:

<<~>> Este simbolo significa mas (+) o menos (-), y si no se introduce ninguno de los dos, el Editor lo interpretara como el signo mas (+).

<<n>> Este simbolo significa que se debe escribir un numero para dar a la orden mas informacion, pudiendo ser cualquier numero entero decimal que este comprendido entre Ø y 65535 ambos inclusive, no pudiendose utilizar comas entre los digitos. En el caso de ser omitido el numero, el Editor interpretara que se trata de un 1, asi como el numero Ø tiene un significado especial en algunas ordenes.

<<CRLF>> Estas iniciales representan las palabras de <CARRIAGE RETURN y LINE FEED>, dado que al introducir una linea de texto se debera pulsar la tecla de Return (<cr>), anadiendo el Editor un avance de linea (Line Feed).

Las ordenes del ED.COM se podran dividir en cuatro funciones, las cuales se detallan a continuacion:

Transferencia de texto

El texto se podra transferir por linea o por bloques

1.- Del fichero original al buffer del Editor.
2.- Del buffer del Editor al fichero temporal (.$$$).
3.- Del fichero original al fichero temporal.
4.- Del buffer del Editor a otro fichero (distinto al original o al temporal)
5.- De otro fichero (distinto al original o temporal) al buffer.

Trabajar con el buffer del Editor

Para localizar el texto en el Editor, se utiliza un caracter puntero (CP) imaginario, el cual le permite lo siguiente:

1.- Insertar texto en el buffer.
2.- Manipular el texto en el buffer en relacion con CP
3.- Controlar el movimiento de CP

Buscar y modificar el texto

Con esta funcion, se podra trabajar en el buffer del Editor en lo siguiente:

1.- Busqueda de un conjunto de caracteres.
2.- Sustitucion de un conjunto de caracteres.

3.- Modificar la colocacion entre los conjuntos de caracteres.
4.- Desplazamiento automatico mientras se busca el texto del fichero original a traves del buffer al fichero temporal.

Combinacion de ordenes

Para permitir varias funciones secuenciales de edicion, y para repetir una cadena de ordenes un numero determinado de veces, se podra teclear en una linea de orden, una cadena de ordenes de ED.

7.1.2 ORDENES DE TRANSFERENCIA DE TEXTO

Orden #A

Esta orden le permite el anadir o copiar una serie de lineas del fichero original al buffer del Editor. En el caso de que se quiera anadir solamente una linea, solo sera necesario el poner A, siendo 1 el numero real de lineas a transferir, pero si por lo contrario, se desea poner el maximo numero de lineas en el buffer, se debera poner la orden #A la cual transfiere el total de las lineas, hasta que el fichero este vacio, o el buffer este lleno. Si se usara la orden ØA, significaria que se desea transferir parte del texto al fuffer del fichero original, hasta que el buffer esta como minimo semilleno.

Orden nW

Esta orden, transfiere una serie de lineas del buffer del Editor a un fichero temporal; comenzando por la primera linea del buffer y escribiendo en el fichero temporal despues de la ultima linea insertada, eliminandose las lineas del buffer una vez que se han escrito en el fichero temporal. Las variaciones de esta orden son las siguientes:

W   Escribe una sola linea
#W   Escribe el buffer del Editor completo
ØW   Escribe hasta que buffer esta como minimo semivacio

Orden E

La mision de esta orden, es indicar al Editor el fin de la sesion de edicion, transfiriendo el texto, y denominando a los ficheros de la siguiente forma:

1.- El total del texto que queda en el buffer es transferido al fichero temporal.
2.- Todo el texto que queda en el fichero original es anadido al fichero temporal.
3.- El fichero original pasa a ser del tipo .BAK
4.- La clase de fichero temporal se cambia al tipo del fichero original.
5.- Se retorna a la peticion de orden del CP/M.

Orden H

Esta orden indica que se dirija al principio del fichero que se esta editando, transfiriendo el texto y los ficheros, que se pueden denominar de las siguientes formas:

1.- Transferir al fichero temporal, todo el texto que queda en el buffer.

2.- Transferir al fichero temporal, todo el texto que queda en el fichero original.
3.- El fichero original se convierte en la clase .BAK
4.- El tipo de fichero temporal se convierte al tipo del fichero original.
5.- Se crea un fichero temporal completamente vacio.
6.- En este paso se esta listo para Editar en el nuevo fichero original.

Existen dos usos principales de esta orden, y son los siguientes:

1.- La orden H se utiliza para poder salvar la edicion que se a hecho hasta ese momento y volver al principio del fichero para continuar editando mas tanto en el fichero temporal como en el original.
2.- Dado que la orden H sirve para salvar lo editado, como hemos mencionado anteriormente, se recomienda el uso de esta orden muy amenudo, para ir salvando la informacion editada en el disquete, dado que el texto que esta en el buffer del Editor se podria perder por cualquier fallo del Computador.

Orden O

Esta orden sirve para borrar el fichero editado, y el texto se transfiere de la siguiente forma:

1.- Se eliminan todos los contenidos del buffer y del fichero temporal.
2.- se retorna al principio del fichero original.

Orden Q

La mision de esta orden es la del abandono de la edicion, sin establecer ningun cambio, teniendo mucho cuidado al usarla, dado que los cambios hechos en el texto se borran cuando se ejecuta esta orden.

Orden R <cr>

Esta orden, lee el fichero creado por la orden X en el buffer del Editor, insertando el contenido total del fichero de transferencia en el buffer inmediatamente despues del caracter puntero. Este fichero se podra leer tantas veces como se quiera durante la sesion de Editor, borrandose cuando uno se salga del programa ED.COM, asi como con las ordenes E, Q o C.

Orden Rnombre de fichero <cr>

Este tipo de orden, lee los nombres del fichero LIB de los ficheros de biblioteca en el buffer del Editor, insertando el contenido completo, inmediatamente despues del caracter puntero, sin alterar dichos ficheros de biblioteca.

Orden nX

La mision de esta orden, es la de escribir o transferir desde el buffer a un fichero temporal denominado X$$$$$$.LIB, pudiendo transferir esas lineas al buffer con la orden R <cr>.

7.1.3 ORDENES DE TRABAJO DEL BUFFER DEL EDITOR

Orden ~B

El desplazamiento del caracter puntero al principio o al final del buffer, se efectua con esta orden, la cual si delante se le pone el signo +, ira al principio, pero si se le pone el signo -, ira al final.

Orden ~nC

Esta orden desplaza el caracter puntero mas o menos <n> caracteres, asi por ejemplo, si se pone <+6C>, indicara que el caracter puntero se desplazara seis caracteres hacia el final del buffer.

Orden ~nD

La mision de esta orden, es la de suprimir un numero <n> de caracteres inmediatamente antes (-) o despues (+) del caracter puntero (CP).

Orden I <cr>

Esta orden sirve para introducir el modo de insertar, aceptando todos los caracteres que se introducen por el teclado, e insertandolos en el buffer del Editor despues del caracter puntero, siendo la forma de salirse de esta orden, el pulsar CONTROL-Z.

Orden Icadena^Z

Al usar esta orden, el sistema le insertara una cadena, que consta de una secuencia de caracteres, en el buffer del Editor, a continuacion del caracter puntero. Esta orden esta destinada para cadenas cortas de caracteres.

Orden Icadena <cr>

La funcion de esta orden, es la de insertar una linea de caracteres en el buffer del Editor siguiendo al caracter puntero (CP). La diferencia entre esta orden y la anterior, es que en esta se le inserta un retorno de carro y un avance de linea.

Orden ~nK

Esta orden, elimina una serie de lineas del buffer, que no se quiere que esten en la version final, por lo cual, dichas lineas no son transferidas al fichero temporal, permaneciendo en el fichero original. La direccion de donde se tienen que suprimir estas lineas, es marcada por los signos de mas (+) o menos (-), con lo cual, si se introduce el signo de +, esta orden eliminara todos los caracteres de la linea despues del caracter puntero, pero si se pone el signo -, esta orden borrara todos los caracteres de la linea antes del caracter puntero.

Orden ~nL

La funcion de esta orden, es el desplazamiento del caracter puntero un numero de lineas "n" hacia adelante o hacia atras, colocandose al comienzo de la linea actual, o de la siguiente si no esta al principio de ella.

Orden ~nP

Desplazara el caracter puntero una pagina y visualizara en pantalla la siguiente pagina, repitiendose un total de "n" paginas, siendo muy util para visualizar todo el texto que esta en el buffer del Editor. Como en ordenes anteriores, el signo mas (+) hara que se deplace hacia adelante, y el signo menos (-), se deplazara hacia atras.

Orden ~nT

Esta orden sirve para ver las lineas de texto que se encuentran en el buffer, asi por ejemplo, si se desea visualizar las cinco lineas antes del caracter puntero, habra que teclear "-5T", y asi mismo, si se desea ver las cinco lineas siguientes al caracter puntero, habra que teclear "+5T" o "5T", sin desplazar el caracter puntero en ninguno de los casos.

Orden ~U

Lo unico que hace esta orden, es la transformacion de las letras minusculas en mayusculas, teniendo que poner +U para comenzar y -U para terminar la conversion.

Orden ~V

La mision de esta linea, es la visualizacion de los numeros de lineas, por lo que si se escribe "V" o "+V", saldran en pantalla todos los numeros de las lineas editadas, pero si se pone "-V" se eliminaran dichos numeros. Una variante a esta orden es la de "ØV", que le indicara en pantalla que parte de memoria se utiliza por el buffer del Editor, y que parte esta disponible para ser utilizada. Dado que saldran en pantalla dos numeros divididos por un barra (/), el primer numero le indicara la memoria disponible para el Editor, y el segundo resena el tamano maximo del buffer, asi pues, si se resta el primer numero del segundo, se tendra el numero de caracteres que se encuentran en ese momento en el buffer del Editor.

Orden :n

Esta orden desplaza un numero "n" de lineas, pudiendose usar como prefijo de una orden haciendo que estas comiencen en un numero especifico de linea. Por ejemplo, si Vd. pone, 15:ØP esta orden hara que se desplace a la linea 15 y visualice una pagina de texto.

7.1.4 COMBINACION DE ORDENES

Una de las ventajas de este Editor, es la combinacion o agrupacion de distintas ordenes, pudiendose poner una detras de la otra, asi por ejemplo:

```
1: *ØA <cr>
1: *B <cr>
1: *T <cr>
1: LINEA 1
1: *
```

Se podra simplificar ahorrando mucho tiempo, si se pone de la siguiente forma:

```
1: *ØABT <cr>
1: LINEA 1
1: *
```

Como en todo esto, hay unas limitaciones en el uso de estas combinaciones, dado que habra que seguir unas reglas cuando se usan varias ordenes en una misma linea, siendo estas las siguientes:

1.- Hay unas ordenes como, E, H, O y Q que deberan ponerse solas en una misma linea para prevenir unas consecuencias de unos errores.

2.- Cuando se usan ordenes que utilicen cadenas, se debera usar CONTROL-Z en vez de <cr> para terminar las cadenas, siendo estas ordenes las F, I, J, N y S.

De todas formas, se debera practicar con este tipo de combinaciones de ordenes, para asi ver sus consecuencias, y su utilidad.

CAPITULO 8

8.1 DUMP.COM

El programa DUMP se utiliza para la visualizacion de un fichero en forma hexadecimal, usado muy frecuentemente por los programadores en lenguaje ensamblador. La forma de llamar a este programa, sera la siguiente:

```
DUMP d:nombre.tip <cr>
```

Este comando, hara visible la representacion en hexadecimal de cada octeto almacenado en el fichero con ese nombre, y en la undad especificada (d:), y en el caso de no especificar la unidad de Disco, el sistema tomara la principal activa.

Otra forma de llamar a este programa, seria la de:

```
DUMP d:*.* <cr>
```

Esta variacion, hace visible en hexadecimal el primer fichero donde coinciden el nombre del mismo *.*, y en la unidad de Disco especificada.

Una representacion del programa DUMP.COM en pantalla, seria la siguiente:

```
A>DUMP B:PROGRAMA.COM <cr>
00 00 3E 09 00 FE 54 2A CD 4E 00 00 2C DF FF EF 00 34
00 10 00 CB BF 09 05 85 87 00 0F 56 EF 00 3E 86 89 DC
00 20 3E 3B BC CB CD 00 87 89 00 45 6E 98 89 00 ED 31
A>
```

Para poder entender esta salida de informacion por pantalla, lo primero que tenemos que aprender, sera el lenguaje maquina de nuestro sistema, significando esta tabla lo siguiente: Los primeros cuatro digitos que aparecen al comienzo de cada linea, es la direccion relativa del primer octeto de esa linea. Los grupos pares de digitos hexadecimales, representan cada uno de los octetos del fichero, siendo un par por cada octeto del mismo.

8.2 SUBMIT.COM

La mision de este programa, es la de dirigir la entrada secuencial y ejecucion de una serie de ordenes de CP/M sin dar ninguna respuesta al operador del Computador, siendo necesario la creacion de un fichero del tipo .SUB usando, por ejemplo, el editor del CP/M, teniendo que

tener, el fichero creado, la lista de ordenes a ejecutar en el orden que se desee que se realicen.

```
SUBMIT nombre <cr>
```

En el momento que ejecutamos este comando, se crea un fichero tipo $$$.SUB en la unidad de Disco que contenga las ordenes listadas en el fichero .SUB, las cuales se ejecutaran antes que las del teclado.

Asi por ejemplo, si se ha creado un programa PRUEBA.SUB que contiene las ordenes:

```
STAT *.BAS <cr>
ERA *.BAS <cr>
DIR *.BAS <cr>
```

Cuando Vd. ejecute este fichero, el dialogo sera el siguiente:

```
A>SUBMIT PRUEBA <cr>
A>STAT *.BAS

RECS   BYTS   EX   D:FILENAME.TYP
2      4K     1    A:NUEVO.BAS
1      2K     1    A:UTIL.BAS
4      8K     1    A:XXXXX.BAS
BYTES REMAINING ON A: 148K
A>ERA *.BAS
A>DIR *.BAS
NO FILE

A>
```

Para poder parar la ejecucion entre alguna de las ordenes, habra que pulsar cualquier tecla, pero antes de que aparezca la peticion de la orden.

```
SUBMIT nombre A B C  <cr>
```

La diferencia entre esta orden de SUBMIT y la anterior, es que en esta orden se pueden incluir lineas de orden incompletas en el fichero .SUB, rellenado el programa SUBMIT la informacion desaparecida usando la A, B, C, etc de la linea de orden introducida.

Para conservar el espacio de los parametros perdidos, se utilizan en el fichero los simbolos $1, $2, $3 $4, etc hasta un maximo de nueve parametros.

Para ver mejor los resultados de este programa SUBMIT, supongamos que Vd. esta editando en un fichero en la unidad A: y quiere copiarlo en la unidad B: examinando el espacio del disco cada vez que se termina una sesion de edicion. En el caso de que no se usase el programa SUBMIT, se deberian introducir las siguientes lineas de orden:

```
A>ED FICHERO.TXT <cr>
A>PIP B:=A:FICHERO.TXT[V] <cr>
A>STAT B:FICHERO.* <cr>
```

En el caso de querer que SUBMIT no haga el trabajo, lo primero que se debera hacer, crear un fichero tipo .SUB, como por ejemplo TRABAJO, conteniendo las ordenes incompletas listadas anteriormente como:

```
ED $1.$2 <cr>
PIP B:=A:$1.$2[V] <cr>
STAT B:$1.* <cr>
```

Por lo que para poder usar el fichero SUB, se debera escribir lo siguiente:

```
A>SUBMIT TRABAJO FICHERO.TXT <cr>
```

La forma con que trabaja esta operacion que parece tan complicada es la siguiente: El programa SUBMIT creara $$$.SUB del fichero TRABAJO.SUB, sutituyendolo cada vez que aparezca el primer parametro, FICHERO, por $1 y el segundo parametro .TXT por $2, iniciandose un arranque caliente del CP/M, buscando el fichero $$$.SUB, y ejecutando las ordenes listadas en este fichero.

8.3 XSUB

El programa XSUB sirve para la automatizacion de entrada de usuario, siendo un subconjunto de SUBMIT. XSUB siempre debera preceder al nombre del programa no pudiendose ejecutar cuando se tiene el control de CP/M (A>), si no que solamente se utiliza en un fichero del tipo .SB como orden de SUBMIT.

XSUB se utiliza en el desarrollo de varios programas y sistemas, comprobando que varios de los programas que Vd, puede comprar usan el XSUB como parte de su entrada. Tambien se puede utilizar esta orden, para suministrar una adicion a su sistema automatico de copias de seguridad que se dicutio anteriormente en la seccicon de SUBMIT.

Asi por ejemplo, si tenemos un fichero &PRUEBA.SUB, que contiene:

```
RUN PROGRAMA.DOC <cr>
XSUB <cr>
BACKUP <cr>
A <cr>
B <cr>
```

Se desarrollara el siguente proceso con la orden SUBMIT.

```
A>SUBMIT &PRUEBA <cr>
A>RUN PROGRAMA.DOC <cr>
...
...
...        (El programa se ejecutara hasta que se termine)
...
...
A>XSUB <cr>
(XSUB ACTIVE)
A>BUCKUP <cr>
DRIVE FOR SOURCE DISKETTE? A <cr>
DRIVE FOR DESTINATION DISKETTE ? B <cr>

PUT BLANK DISKETTE IN B AND PRESS RETURN WHEN READY <cr>
```

La esplicacion de lo que pasa en este ejemplo, es lo siguiente:

Desde el momento de que Vd. introduce la orden de SUBMIT hasta que le da la peticion de poner un disquete en blanco en la unidad B: y pulsar <cr>, la Computadora realizara automaticamente todas las operaciones suministrando XSUB a A <cr> y B <cr> en respuesta al programa de BACKUP.

8.4 MENSAJES DE ERROR

En el CP/M existen una serie de mensajes de error, que le indican al operador lo que esta mal y en donde, asi mismo, cada programa en este Sistema Operativo tiene su serie de mensajes, los cuales los discutiremos a continuacion.

8.4.1 Errores en el BIOS

Indica que existe un error de entrada o salida en acceso al disco por lo que le saldra la pregunta de: RETRY, IGNORE or CANCEL, por lo que Vd. debera responder con una R, I o C, significando que la operacion se vuelva a entrar, ignorar, o cancelar. Ademas de todo esto, hay tres tipos de errores dentro del BIOS:

1.- BAD SECTOR, que indica que se esta usando un disquete sin formatear, que hay un sector mal, o que el disquete se haya colocado al reves.
2.- SELECT, indicando que no encuentra el dispositivo que se ha seleccionado.
3.- R/O, indica que se ha colocado un disquete equivocado o que el disquete tiene puesta la proteccion de escritura.

Existen ademas los siguientes mensajes de BIOS para errores:

DRIVE d:DISK MISSING, indicando que el BIOS esta intentando acceder a esa unidad, y falta por poner el disquete.

FAILED TO LOAD CP/M, indica que se ha producido un error en arranque en caliente cuando un sector del CCP o del BDOS no se ha leido correctamente.

8.4.2 Errores en el Programa STAT

1.- ABORTED significa que se termino la orden de STAT
2.- BAD DELIMITER se debera verificar la coplocacion de la puntuacion.
3.- INVALID ASSIGNMENT indica que no se puede asignar ese dispositivo de la forma especificada.
4.- INVALID FILE INDICATOR significando que no se pueden asignar los nombres de ficheros para utilizarlos de la forma especificada.
5.- TOO MANY FILES, significa que como el programa STAT tiene un limite de ficheros que puede clasificar, trata de usar referencias ambiguas de ficheros.
6.- INVALID DISK ASSIGNMENT, indicando que no se puede asignar el disco de la forma especificada.

8.4.3 Errores en el programa PIP

1.- DISK READ ERROR: error de lectura
2.- DISK WRITE ERROR: error de escritura
3.- VERIFY ERROR: error de verificacion
4.- NOT A CHARACTER SINK: indica que no se pueden enviar los caracteres.
5.- READER STOPPED: paro de la lectora
6.- NOT A CHARACTER SOURCE: indica que no se pueden tomar los caracteres.
7.- ABORTED: fin del proceso.
8.- BAD PARAMETER: los parametros [] son incorrectos.
9.- INVALID FORMAT: formato no correcto.
10.- NO DIRECTORY SPACE: no hay espacio para ficheros en el directorio.
11.- CANNOT WRITE: no puede el PIP escribir.
12.- CANNOT READ: no puede el PIP leer.
13.- CANNOT CLOSE FILE: examinar el disquete.
14.- INVALID SEPARATOR: examinar puntuacion

8.4.4 Errores en el Programa SUBMIT

1.- ERROR ON LINE: linea de orden incorrecta del fichero SUB
2.- NO .SUB FILE PRESENT: no puede encontrar el fichero .SUB
3.- DISK WRITE ERROR: disco protegido contra escritura.
4.- COMMAND BUFFER OVERFLOW: orden muy larga.
5.- COMMAND TOO LONG: orden con mas de 127 caracteres.
6.- DIRECTORY FULL: directorio lleno

8.4.5 Errores en el Programa XSUB

1.- XSUB ALREADY PRESENT: hay una orden innecesaria XSUB en el fichero .SUB

8.4.6 Errores en el Programa ED

1.- BREAK "X" AT "C": "X" se refiere a uno de los seis simbolos:
- # buscar fallo
- ? orden desconocida
- Ø fichero no puede ser encontrado
- > el buffer esta lleno
- E la orden fue abortada
- F el disco esta lleno

CAPITULO 9

9.1 LENGUAJE ENSAMBLADOR ASM.COM

El programa ASM.COM es un compilador que le convierte los programas en codigo fuente, escrito en lenguaje ensamblador, en programas objeto con formato hexadecimal. Esta forma de conversion, es un paso para llegar a un programa en lenguaje maquina, que es el que el sistema puede interpretar. La orden LOAD y GENCMD, le realizara la conversion final, creando un fichero para el programa en lenguaje maquina, partiendo del programa en hexadecimal. Asi mismo, la orden DDT es un programa de depuracion, para poder corregir y localizar errores de los programas en lenguaje maquina.

Cualquier programa escrito en lenguaje ensamblador, pasara dos veces por el proceso del Computador, recordando que esta utiliza 1 y 0 como elementos basicos de datos, siendo estos, o bien datos o instrucciones de la Computadora. Dado que no conviene pensar en terminos de numeros como por ejemplo: 00110011 y 01100011, se tendra que usar notacion hexadecimal para poder referirse a los ocho bits a la vez, llamando a estos grupos de ocho bits "OCTETOS" (Bytes) de informacion, siendo estas representaciones de octetos hexdecimales, como por ejemplo, 0A2, 56 y 0F, recordando que la Computadora solo utiliza notacion binaria.

Dado que la representacion hexadecimal es bastante complicada, se tendra que asignar nombres a las instrucciones maquina y los datos que representan, siendo los nombres los componentes del lenguaje de la Computadora, conocidos como "Lenguaje Ensamblador", llamando a esos nombres, intrucciones "Nemotecnicos", siendo este una representacion de una instruccion de maquina de CPU.

Los nemotecnicos del lenguaje ensamblador de la CPU 8085 son casi identicos a los del 8080, pero sin embargo, los del Z80 son completamente diferentes, significando lo mismo las intrucciones maquina.

9.2 COMO ENSAMBLAR UN PROGRAMA

El ensamblador de CP/M, ASM.COM, convierte los programas fuentes en lenguaje ensamblador a codigo objeto para poder ser ejecutado por la Computadora.

El codigo fuente del lenguaje ensamblador, se podra generar escribiendolo en un fichero utilizando un Editor, como el que Vd. tiene en su CP/M, ED.COM descrito anteriormente.

9.2.1 Ordenes de ASM.COM

Para poder empezar, y como hemos mencionado al principio de esta seccion, se debera haber creado un fichero fuente, con lo que Vd. podra llamar al ensamblador tecleando:

```
A>ASM FICHERO.OPT <cr>
```

El nombre del fichero debera ser del tipo ASM para que sea valido para el CP/M. El tipo OPT (opcion) en la linea de orden se refiere a una serie de tres opciones que especificaremos, y no a un tipo de fichero. Estos tres tipos son los siguientes:

1.- FICHERO.ASM que es el dispositivo que contiene el fichero fuente.
2.- FICHERO.HEX es el dispositivo que recibira el fichero hexadecimal
3.- FICHERO.PRN es el dispositivo que recibira el fichero de impresion, listando el programa con los mensajes de error.

Tambien se puede introducir la opcion de asignar la unidad de disco, usabdo las letras A y B, para representar las dos unidades que Vd. puede tener. Asi mismo, hay tambien otras letras de opcion que son la X y la Z, teniendo significados especiales detallados a continuacion:

La letra Z se podra utilizar como opcion en los ficheros PRN o HEX para evitar la generacion de ese fichero.

La letra X se utiliza en el fichero PRN para poder visualizar el listado en la pantalla, en vez de almacenarlo en un fichero.

Asi pues, si se pone lo siguiente:

```
A>ASM PRO.AAZ <cr>
```

Se estara indicando al ensamblador, que ensamble un fichero denominado PRO.ASM que se encuentra en la unidad A: con un fichero objeto llamado PRO.HEX, tambien en la unidad A: y no generar el fichero de impresion.

Una ves ensamblado el programa PRO.ASM, se obtendran los siguientes ficheros:

PRO.ASM	Fichero fuente
PRO.HEX	Fichero con el programa ensamblado
PRO.PRN	Fichero para poder listar el programa

9.2.2 Ficheros del ASM.COM

El programa ensamblador, crea un fichero del tipo HEX, conteniendo las representaciones hexadecimales de las intrucciones del fichero ASM.

Un ejemplo de un fichero en HEX es el siguiente:

```
:020100003E00BF
:0000000000
```

A continuacion se muestra este programa en lenguaje ensamblador que crea esta informacion:

```
ORG   0100        ;comienzo del programa en la direccion 0100 hex
MVI   A,0         ;colocacion de 00 hex en el registro A
END               ;fin del programa
```

El fichero tipo .PRN convinara la informacion de los dos conjuntos anteriores, teniendo el programa creado originalmente y el ensamblado:

```
0100            ORG    0100      ;comienzo del programa en 0100 hex
0101 3E 00      MVI    A,0       ;coloca 00 hex en el registro A
0102            END              ;final del programa
```

9.3 SENTENCIAS DEL LENGUAJE ENSAMBLADOR

Cada sentencia en el lenguaje ensamblador, esta compuesta por una serie de campos, con una variable entre uno y cinco espacios, siendo un campo, un grupo de caracteres y separados los campos uno del otro por espacios o caracteres de tabulacion. Los caracteres de tabulacion (TAB) crearan un programa fuente mas intelijible, dado que la separacion entre ellos esta alineada. Es aconsejable que la tabulacion este puesta en las columnas, 1, 9, 17, 25, 33, etc, osea cada ocho columnas.

El formato de la sentencia del lenguaje ensamblador es el siguiente:

LINEA#	ETIQUETA	NEMOTECNICOS	OPERANDOS	COMENTARIOS

Numeros de lineas (LINEA#)

El numero de linea, es un numero decimal entero puesto al comienzo de una linea del codigo fuente, que van indicando las lineas de edicion. Algunos Editores insertan estas lineas automaticamente, y en el caso del ED.COM no lo hace, por lo que se debera introducir desde el teclado, aun que esta numeracion es opcional dado que el ensamblador no la reconoce.

Etiquetas

La etiqueta es un identificador que sirve para representar una direccion o valor, teniendo una longitud de 16 caracteres. El primer caracter debe ser siempre una letra, siendo el resto letras o numeros.

Las etiquetas son opcionales para todas las sentencias, excepto para aquellas que lo utilicen en el campo del operando como con los directivos EQU y SET. Normalmente, una etiqueta particular aparecera en su campo como una sola sentencia, pero puede aparecer en el campo del operando de muchas sentencias.

Nemotecnicos

El unico que no es opcional en el lenguaje ensamblador es el campo de los nemotecnicos, y debera estar incluido en la

sentencia, conteniendo, o bien un nemotecnico de una instruccion, o el nombre de un directivo del ensamblador.

Dado que estos nemotecnicos son el corazon del leguaje ensamblador, habra que estar muy familiarizado con todos ellos antes de iniciarse a la programacion en este tipo de lenguaje.

Operandos

Los operandos, son requeridos mas de una vez en las instrucciones del lenguaje ensamblador, mientras que hay otras intrucciones que no los requieren. Un operando puede ser una constante, una etiqueta o una expresion.

Comentarios

El comienzo del campo de los comentarios, debera empezar siempre con un punto y coma (;), siendo este campo opcional a la hora de editar el programa y es ignorado por el ensamblador. La practica de incluir los comentarios, es muy aconsejable, dado que le orientan en la forma de como se interpreta el programa y lo que hace en ese momento, siendo una documentacion muy esencial y practica.

9.3.1 EXPRESIONES Y CONSTANTES

Los campos de un operando, pueden estar ocupados por un titulo, una constante o una expresion. Dado que los titulos se explicaron anteriormente, a continuacion se explicaran las expresiones y las constantes.

Las constantes

Una constante numerica, es un numero fijo en uno de los cuatro sistemas de numeracion, siendo los sistemas de numeracion los siguientes:

CONSTANTE BINARIA es una secuencia de digitos 0 y 1 seguidos por la letra B que indica que es un numero binario.
CONSTANTE OCTAL es una secuencia de digitos comprendidos entre el 0 y el 7 seguidos de la letra Q u O con lo que indicara que ese numero es Octal.
CONSTANTE DECIMAL es una secuencia de digitos comprendidos entre el 0 y el 9 seguidos de la letra D para indicar que se trata de un numero decimal, asi mismo tambien se podra omotir la letra D, dado que ASM lo reconocera como numero decimal.
CONSTANTE HEXADECIMAL es una secuencia de digitos comprendidos entre el 0 y 9 asi como las letras A a la F seguidos de la letra H que le indica al ensamblador de que es un numero hexadecimal.

Expresiones

Una expresion puede ser una combinacion de constantes, titulos, operadores aritmeticos, operadores logicos y parentesis, reduciendose a un valor simple cada expresion durante el ensamblado.

Los Operadores Aritmeticos son los que pueden hacer una operacion simple cuando evalua una expresion para poder determinar su valor. Se podran utilizar los operadores aritmeticos de sumar, restar, multiplicar y dividir, como por ejemplo A+B es la suma de A y B, o A-B seria la resta de A menos B, etc.

Los Operadores Logicos son lo mismo que los operadores aritmeticos, realizando el ensamblador operaciones booleanas, pudiendose usar los siguientes operadores:

NOT B es el complemento bit a bit de B
A AND B es la operacion logica AND bit a bit de A y B
A OR B es la operacion logica OR bit a bit de A y B
A XOR B es la operacion logiga exclusiva de OR bit a bit de A y B

9.4 DIRECTIVOS DEL LENGUAJE ENSAMBLADOR

Los directivos del ensamblador controlan el proceso y afectan al resultante del codigo maquina, debiendose colocar estas sentencias en el programa fuente de la misma forma que las del lenguaje fuente. El nombre de la sentencia directiva, va en el campo nemotecnico de la sentencia.

9.4.1 Directivos DB, DW, DS

Estos directivos son utilizados para inicializar las areas de almacenamiento en memoria.

DB, este directivo define el octeto inicializando un area octeto a octeto en posiciones consecutivas de memoria

DW, este directivo define la palabra, inicializando en un area dos octetos a la vez, o sea en pares consecutivos de posiciones de memoria.

DS, este directivo define el almacenamiento, reservando un area de memoria de tamano especificado.

9.4.2 Directivos ORG, END, EQU

ORG: Este directivo indica al ensamblador la direccion de memoria que se va a utilizar por las sentencias que siguen, siendo opcionales los comentarios y pudiendose utilizar en cada programa, mas de un ORG.

END: Este directivo, indica al ensamblador, que no hay mas sentencias en el programa fuente, siendo opcional.

EQU: Este directivo, asigna valores o expresiones a un titulo, siendo su formato:

```
Titulo          EQU expresion          :comentario
```

La expresion puede ser cualquier numero valido, direccion, constante o expresion, siendo necesarios en una sentencia EQU el titulo y la expresion.

Existe una variante del directivo EQU, la cual es el directivo SET, siendo la diferencia en que permite reasignar el valor a un titulo, siendo el mismo formato para los dos.

9.4.3 Directivos IF y ENDIF

El formato de estos dos directivos es el siguiente:

```
Titulo          IF          Expresion            ;comentario
                            .........
                            .........
                            serie de sentencias
                            .........
                            .........
Titulo          ENDIF                            ;comentario
```

El lenguaje ensamblador, evaluara la sentencia IF, y si el valor de la expresion es cero, el ensamblador ignorara las sentencias que estan entre IF y ENDIF, pero si el valor es distinto a cero, las ensamblara correctamente. El uso de esta sentencia IF se denomina "Ensamblamiento Condicional", realizandose si cumplen estas condiciones descritas anteriormente.

En el caso de no estar muy esperimentado con el lenguaje ensamblador, le recomendamos que se lea algun libro sobre las direcciones de la CPU Z80 que hay en el mercado para poder conocer mas a fondo el significado de sus nemotecnicos.

CAPITULO 10

10.1 DDT

Esta orden es parte del ensamblador, dado que es una herramienta de depuracion dinamica, usandose para examinar y depurar programas en lenguaje maquina. Este tipo de herramienta para el lenguaje ensamblador, la podemos utilizar para las siguientes funciones:

1.- Se podra cargar un programa ensamblado en memoria.
2.- Hacer en un programa de lenguaje màquina cambios sencillos.
3.- Asi mismo localizar errores en un programa de lenguaje maquina.
4.- Efectuar correcciones.
5.- Instalacion de rutinas, para poder comandar perifericos.
6.- Cambio o modificacion de para metras de Disco.
7.- Modificar y examinar los contenidos en memoria.
8.- Introduccion en una linea el codigo del lenguaje ensamblador.
9.- Desensamblar una parte del programa.
10.- Modificar y examinar los contenidos de los registros internos.
11.- Introducir posiciones donde se detiene un programa y examinar lo que pasa.
12.- Seguir lo que ocurre en la memoria durante la ejecucion de un programa, asi como en los registros internos.

Hay dos formas de poder cargar este programa (DDT.COM) en memoria, una tecleando:

```
DDT <cr>
```

y la otra, donde ademas de cargar el programa, se carga el fichero que ha de ser examinado, y para ello se debera teclear:

```
DDT d:FICHERO.TIP <cr>
```

El tipo de fichero, debera ser COM o HEX para poderlo modificar o extenderlo. Una vez que DDT este en memoria esta esperando unas ordenes para poder usar sus propiedades.

10.1.1 Ordenes de DDT.COM

Casi todas las ordenes de DDT requeriran informacion adicional para que puedan ser utilizadas, siendo a menudo una direccion de memoria o un valor hexadecimal.

Orden A# <cr>

La funcion de esta orden es la de introducir las instrucciones del lenguaje ensamblador en la direccion de memoria especificada en la orden, sin que deba haber ningùn espacio entre la letra A y el numero hexadecimal. Una vez entrada la orden, el DDT visulizara la direccion de memoria que hemos especificado, esperando que se pongan las

intrucciones deseadas las cuales deberan ser un nemotecnico u operando valido. Estos dos deberan estar separados por un espacio y cada intruccion se terminara con un <cr>, apareciendo la siguiente direccion disponible para proseguir metiendo intrucciones como hemos espresado anteriormente.

En el caso de que se desee finalizar, se podra hacer de dos formas, o introduciendo un punto (.) seguido de un <cr>, o con solo pulsar un <cr>.

Orden D <cr>

La orden D es muy util para poder visualizar parte de la memoria de nuestro Computador, existiendo tres formas de hacer esta visualizacion en pantalla. La primera, es entrando solamente la letra D con un <cr> con lo cual, le indicamos al programa DDT que nos visualice la posicion actual de la memoria, apareciendo en la parte izquierda de la pantalla, la direccion de memoria en cada linea, seguida de 16 numeros hexadecimales de los octetos que estan o comienzan en esa direccion, a continuacion se mostrara 16 caracteres en ASCII de los cuales algunos seran puntos los cuales representan a caracteres que no se pueden identificar, siendo la equivalencia de los 16 numeros hexadecimales o octetos.

D# <cr> es la segunda forma de poder entrar este tipo de orden, diferenciandose de la anterior, en que en esta se le puede especificar la direccion de memoria de comienzo para que sea visualizada. Una de las condiciones para que todo esto funcione adecuadamente, es que la direccion de memoria que se va a introducir sea multiplo de 16, pues de lo contrario no se representaran las 16 posiciones.

D#,# <cr> sera la tercera forma en este tipo de orden, siendo esta muy parecida a las anteriores, con la unica diferencia que en este caso se le indica al Computador el comienzo y final de lo que se desea visualizar, por lo que se debera introducir las posiciones de memoria las direcciones donde se quiere empezar y la final, las cuales deberan terminar siempre en Ø, dado que de lo contrario, no apareceran los 16 numeros hexadecimales

Orden F#,#,# <cr>

Esta orden se untiliza para llenar un bloque de memoria con un octeto, consistiendo en poner un caracter de valor nulo en una seccion de memoria antes de hacer un nuevo trabajo.

Para la introduccion de esta orden, se necesitara poner tres numeros que sigan a la letra F, siendo el primer numero la direccion donde finaliza el bloque de memoria, y el tercero sera el octeto a insertar en dicho bloque, como por ejemplo:

```
-FØ1ØØ,Ø1CØ,Ø7Ø <cr>
```

En este ejemplo, esta orden colocara el valor hexadecimal 7Ø en todas las direcciones de memoria entre Ø1ØØ Hex y Ø1CØ Hex ambas inclusive. Habra que tener mucho cuidado con esta orden, dado que se puede llenar posiciones de memoria que son utilizadas por el CP/M o DDT, las cuales generalmente son las 256 primeras posiciones de memoria y los ultimos 6K o 1ØK de la memoria.

Orden G <cr>

Esta orden indica al programa DDT el comienzo de una ejecucion de instrucciones de memoria, a partir de la direccion contenida en el contador de la CPU. Existen cuatro variantes de esta orden, los cuales indican diferentes comienzos y diferentes formas de ejecucion.

Estas varientes son las siguientes:

G <cr>

Como hemos mencionado anteriormente, esta orden indicara que comience la ejecucion de las intrucciones de memoria, a partir de la direccion contenida en el contador de la CPU. Muy raras veces se utiliza esta orden para depurar un programa sin especificar una direccion final, dado que el programa DDT no tiene control sobre este tipo de ejecucion sin poder pararlo, a no ser que se le haya introducido una instruccion de RST.

G# <cr>

Esta variante, indica el comienzo de la ejecucion en una direccion distinta a la del contador del programa, teniendo que poner a continuacion de G un numero hexadecimal que indica la posicion de memoria de comienzo. A igual que pasa en la variante anterior, DDT no podra recuperar el control.

G#,# <cr>

La tercera variante de esta orden, especifica el comienzo y final de las instrucciones a ejecutar, y para ello habra que poner las dos direcciones separadas por una coma a continuacion de le letra G siendo estas dos cifras numeros hexadecimales.

Una forma de poder retornar al programa DDT, es el introducir en esa orden, dos direcciones, que cuando se encuentran, retornan al DDT teniendo que expresarla como G#,#,# <cr>.

G,# <cr>

En esta orden, se puede omitir la direccion de comienzo, indicando al DDT que utilice el contador actual del programa, siendo muy util para continuar la ejecucion despues de encontrar un punto de ruptura.

Orden H#,# <cr>

Este tipo de orden aritmetica Hexadecimal, le calcula la suma y la diferencia de dos numeros decimales, convirtiendose primero a valores de 16 bits antes de que se efectuen los calculos, siendo el primer numero la suma, y el segundo la diferencia entre los dos.

Orden L

En este tipo de orden hay dos variantes, la primera se entrara como L <cr> sirviendo para desensamblar una porcion de memoria, listando el contenido de la ultima direccion. En el caso de que el programa DDT no sepa como visualizar un valor en hexadecimal encontrado en el lenguaje ensamblador, aparecera un mensaje como: ??=##, en donde ## corresponde al octeto hexadecimal. Cada vez que se usa esta orden, se visualizaran las once intrucciones siguientes, en donde los operandos que tienen valores numericos, los mostrara en numeros hexadecimales, asi como no visualizara los titulos ni los simbolos.

La segunda variante de esta orden es la que se entra como, L#,# <cr> la cual funcionara identicamente como la anterior, con la unica diferencia que se le pedira que muestre el principio de una direccion y el final. Estas dos direcciones deberan ir separadas por una coma en la orden L.

Orden M#,#,# <cr>

Esta orden sirve para mover parte del contenido de memoria, constando de tres numeros, donde el primero es la primera posicion a desplazar, el segundo es la ultima posicion, y el tercero es la primera posicion del nuevo bloque de memoria especificado.

En el caso de que apareciese una interrogacion despues de introducir esta orden, significara que por lo menos uno de los numeros especificados no es una direccion hexadecimal, y tambien si la ultima posicion de memoria a desplazar es menor que la posicion inicial.

10.2 CARGAR Y CREAR UN PROGRAMA EJECUTABLE

Orden LOAD

La funcion de sta orden es esclusivamente para tomar un fichero tipo HEX, creado por el ensamblador, y convertirlo en un fichero ejecutable tipo COM el cual contiene el codigo maquina del fichero HEX pero lo comienza en la direccion 0100 hex.

La forma de entrar esta orden, es la siguiente:

```
A>LOAD B:NOMB <cr>
FIRST ADDRESS            0100
LAST ADDRESS             0355
BYTES READ               0256
RECORDS WRITTEN            02
A>
```

En el caso de que el fichero sea grande, el disco sera activado durante un tiempo mientras LOAD hace su funcion.



11.1 PROGRAMAS DE UTILIDAD EN CP/M

En este Capitulo le vamos a presentar algunos Programas de utilidad para CP/M, los cuales son de gran ayuda para el usuario. Existen varias casas destinadas a la venta de programas de utilidad, los cuales son un poco mas sufisticados que los que normalmente trae el Disco original de CP/M.

11.1.1 Programa FORMAT.COM

Para poder almacenar informacion en los discos de Amstrad, lo primero que deberemos hacer sera el formatear el disco, lo cual significa el dividir el disco como si de un libro se tratase, y de esa forma la Computadora sabra donde poner la informacion.

Un ejemplo de como formatear un disquete virgen seria el siguiente:

```
A>FORMAT <cr>
DISK TO FORMAT? (A,B,C,D) A <cr>
PRESS RETURN TO BEGIN FORMATTING <cr>
DISK FORMATTED. MORE (Y/N) N <cr>
A>
```

Vd. podra notar que la sintaxis de los mensajes que aparecen en este programa pudieran variar de una version de CP/M a otra, pero lo que si es cierto es que variaran muy poco en su contenido, siguiendo el programa los siguientes pasos.

1.- Ejecuta el programa despues de pulsar el <cr>
2.- El programa le pidira en que unidad de disco esta el disquete a formatear.
3.- Indicar al programa que empiece a formatear.
4.- Indicar al programa si deseamos finalizar esta sesion, o deseamos fomatear otro disquete.

Este programa empieza por desplazar la cabeza magnetica de su disco al primer sector de la primera pista, escribiendo alguna informacion ficticia. una vez que se ha escrito cada sector en una pista, la cabeza avanza a la siguiente pista siguiendo de esta forma hasta que el disquete se ha formateado completamente.

11.1.2 Programa MOVCPM.COM

La accion de este programa le permite el desplazamiento del sistema CP/M a una posicion de memoria diferente a la actual, que normalmente lo traslada a la parte inferior de la misma. Todo esto se suele hacer cuando tenemos programas muy grandes y necesitamos parte de la memoria

que esta ocupada por el sistema de CP/M y sus programas transitorios. El CP/M se podra colocar en cualquier posicion de memoria que este situada sobre el limite de 256 bytes, siendo el tamano de su parametro entre 64 a 179 dentro del area disponible de 256 bytes.

Hay dos formas de poder ejecutar este programa, la primera desplazara CP/M y lo ejecutara inmediatamente usando toda la memoria existente, y para ello se debera teclear:

```
A>MOVCPM <cr>
```

La segunda forma es el poner a continuacion el numero decimal de kilo-octetos de memoria que se desea que el CP/M reconozca, siendo muy util cuando se quiere reservar sitio en la misma, para ello se debera teclear lo siguiente:

```
A>MOVCPM# * <cr>
```

Donde # es el numero de kilo-octetos de memoria que se desea que el CP/M reconozca.

11.1.3 Programa SYSGEN.COM

El proposito de este programa es el colocar una copia del sistema operativo en un disquete, denominado generacion del sistema. Asi mismo, escribira el resultado de un comando de MOVCPM en las pistas de un disco del sistema.

La generacion del sistema se podra realizar de una de estas tres maneras diferentes:

1.- El SYSGEN se puede usar solamente para copiar sin ningun cambio el sistema de un disquete y colocarlo en uno nuevo.
2.- Se podra usar, utilizando el MOVCPM y despues el SYSGEN para insertar el nuevo sistema en un disquete.
3.- Este punto es un poco mas complicado, dado que utilizando el MOVCPM y almacenarlo en un disquete para luego volverlo a cargar para poder ser modificado por el programa DDT, y el resultado usando el programa SYSGEN salvarlo en un disquete.

En esta seccion no entraremos en la modificacion del sistema, por lo que solo decribiremos el uso solamente del SYSGEN junto con la utilizacion de MOVCPM. El uso del SYSGEN no afecta para nada a los ficheros de los disquetes fuente o de destino.

Lo que le aparecera en pantalla cuando se trata de copiar el sistema de un disquete a otro, sera lo siguiente, aunque puede variar un poco con respecto a su version de CP/M:

```
A>SYSGEN <cr>
SYSGEN VER 2.2
SOURCE DRIVE NAME (OR RETURN TO SKIP) A <cr>
SOURCE ON A:, THEN TYPE RETURN <cr>
FUNTION COMPLETE

DESTINATION DRIVE NAME (OR RETURN TO REBOOT) B <cr>
DESTINATION ON B:, THEN TYPE RETURN <cr>
DESTINATION DRIVE NAME (OR RETURN TO REBOOT) <cr>
A>
```

A continuacion, examinaremos paso a paso lo que pasa en este procedimiento.

Lo primero que se hace sera el cargar el programa SYSGEN para poderlo ejecutar. A continuacion le pidira la unidad fuente de donde se ira a copiar, escribiendo la unidad A, preguntando a continuacion a que unidad de disco se destina la copia del sistema, con lo que le ordenaremos que sea la unidad B, siendo en este momento cuando las dos unidades enpiezan a funcionar para efectuar las copias. Una vez finalizada esta operacion, se pulsara la tecla de <cr> para hacer un arranque en frio (reboot) y volver al control de CP/M.

El porque es necesario el uso del programa SYSGEN se lo aclararemos en pocas palabras. En primer lugar las dos primeras pistas (pistas Ø v 1) del disquete de CP/M no contienen ficheros, siendo este espacio de 6656 octetos y reservado para los cargadores de arranque en frio asi como el sistema operativo, asi pues estos programas no se almacenan como ficheros sin aparecer en el directorio y no son accesibles como tales programas.

Siempre que se copia un disquete, las pistas Ø y 1 se saltan aunque no contengan el cargador o los programas del sistema, no siendo necesario el almacenar estos programas en cada disquete dado que el arranque en frio se puede hacer desde un solo disquete.

Cuando se hace una copia de un disquete entero, hay muchos que contienen el sistema operativo grabados en el original, por lo que los programas del sistema se copiaran a la par que los demas ficheros, asi como el arranque en frio (boot). La razon por la que se necesita el programa SYSGEN en estos casos, es para poder mover o colocar el sistema operativo en caso de necesitar espacio para la ejecucion de programas grandes.

Hay un punto muy importante en una generacion del sistema, y es que el MOVCPM solo desplaza el esqueleto del sistema operativo, y en este caso cualquier impresora especial u otros cambios que se hubiesen hecho en el BIOS no son desplazados por el CP/M. En el caso de ser utilizado el MOVCPM y despues la impresora o cualquier otro dispositivo que se quiera utilizar; y no funcionen, seria necesario, en este caso, el anadir controles especiales a estos dispositivos.

11.2 ESTRUCTURA TECNICA DEL CP/M

Esta seccion sera de gran utilidad para el programador que quiera conocer la efectividad del CP/M.

Para que entienda Vd. la estructura del CP/M, le diremos que el sistema ocupa la parte superior de la memoria, y los programas de usuario y de datos pueden ocupar el area de memoria entre Ø1ØØ hex y el la parte inferior.

El mapa de memoria de un sistema CP/M ser el siguiente:

```
                     ---   -------------   FFFF
                      I    I           I
                      I    I   BIOS    I
                      I    I           I
                      I    -------------   xx00 + 1600
                      I    I           I
      CP/M            I    I   BDOS    I
                      I    I           I
                      I    -------------   xx00 + 0800
                      I    I           I
                      I    I    CCP    I
                      I    I           I
                     ---   -------------   xx00
                      I    I           I
                      I    I    TPA    I
     Programa         I    I           I
                      I    -------------   0100
    Transitorio       I    I           I
                      I    I   BASE    I
                      I    I           I
                     ---   -------------   0000
```

El procesador de ordenes de consola, se le denomina modulo CCP interpretando las ordenes que son introducidas por la misma siendo relevante solamente cuando aparece en pantalla la A> o la de otra unidad. Cuando se introduce esta orden, se almacena primero en la posicion 0800 - 00FF hex, y si el CCP no reconoce la orden, busca en el directorio del disco un fichero tipo .COM cuyo nombre tenga los primeros ocho caracteres identicos. Una vez encontrado, se carga en memoria a partir de la direccion 0100 hex y su ejecucion se pasa a 0100 hex cuando el fichero este completamente cargado llamando a ese fichero orden transitoria.

Casi toda la actividad de la unidad de disco, asi como la mayoria de la consola se encarga la seccion del CP/M llamado el BDOS, el cual no se le puede acceder a traves de las ordenes de la consola.

La CPU para la cual fue escrito el CP/M consta de siete registros de almacenamiento interno de 8 bits y ocho bits de estado, seis de estos registros se emparejan en tres registros de 16 bits, siendo su utilizacion normal en lenguaje ensamblador de la siguiente forma:

REGISTRO	UTILIZACION
A (acumulador)	Caracter de ocho bits de entrada/salida o registro de manipulacion de datos.
B	Registros de 8 bits usados separadamente.
C	Emparejados un registro de 16 bits.
D	Registros de 8 bits usados separadamente
E	Emparejados un registro de 16 bits.
H	Registros de 8 bits usados separadamente
L	Emparejados un registro de direccion de memoria de 16 bits.
Estado	Senalizacion de 8 bits usados para indicar arrastre, paridad, cero, etc.

Dado que Digital Research utiliza de una forma muy peculiar los registros para las funciones del BDOS, su utilizacion sera la siguiente:

A (acumulador)	Cualquier valor de ocho bits del BDOS
B	Este registro no se usa en el BDOS para llamadas.
C	Numero de funciones del BDOS a utilizar
DE	Posicion de memoria variable de 16 bits
HL	Valor de 16 bits del BDOS

El BIOS es un sistema muy particular de entrada/salida en el CP/M el cual le indica cualquier anomalia en las entradas o salidas del sistema.

Las funciones que realiza el BIOS son dependientes del Hardware que tiene el Computador, indicando como acceder a distintos dispositivos que forman el sistema de calculo, como por ejemplo el movimiento de la cabeza de una unidad de disco. Cualquier error en esta seccion, puede ser causante de que la Computadora no funcione o tenga fallos.

Para esta version de CP/M el BIOS empieza en la posicion de memoria 1600 hex comenzando con una serie de instrucciones de bifurcacion, debiendo estar colocadas correctamente y estando todas presentes para que el BIOS funcione correctamente.

CAPITULO 12

12.1 LEGUAJE LOGO

Mas que cualquier otro lenguaje, el LOGO se creo para desmitificar y perderle el miedo a las Computadoras, asi como en su programacion, haciendolo accesible a todo el mundo. La variedad de aplicaciones que tiene este tipo de lenguaje puede ir desde la ensenanza escolar pasando por la investigacion hasta la creacion de graficos de alta resolucion. Una aplicacion muy importante, fue la aplicacion del LOGO en la ensenanza de los deficientes mentales permitiendo que estas personas puedan controlar el mundo de la informatica.

El lenguaje LOGO fue desarrollado en la decada de los anos 70 por una serie de cientificos bajo la batuta del Dr. Seymour Paper que ademas se le incluyo el uso de la "Tortuga" para que los ninos pudiesen programar graficos en una Computadora, asi como tambien se pudiese usar por personas disminuidas fisicamente.

12.2 COMO CARGAR EL LENGUAJE LOGO

Como Vd. ha podido comprobar, el lenguaje LOGO se en cuentra en su disquete original en la cara dos, por lo que debera colocar ese disquete por esa cara, o si lo ha copiado en otro disco de seguridad utilizar ese mismo. Amstrad denimina al LOGO como Dr. LOGO, asi pues inicialice el sistema pulsando simultaneamente las teclas [CTRL], [SHIFT] y [ESC]. A continuacion teclee ¦CPM y automaticamente el Dr. LOGO sera leido del disco y cargado en memoria, saliendo en su pantalla el siguiente mensaje:

```
                Welcome to

           Amstrad LOGO V1.1
  Copyright (c) 1983/4, Digital Research
        Pacific Grove, California

Dr. Logo is a trademark of Digital Research

        Serial No. 6002-1232-123456

               Please Wait
```

Este mensaje le indicara que se esta cargando el LOGO por lo debera esperar a que termine la operacion de carga con lo cual este mensaje desaparecera, y en sustitucion tendra en su pantalla el signo de interrogacion (?) siendo esta indicacion la del control del Dr. LOGO y que esta esperando las ordenes que Vd. quiera introducirle por el teclado.

12.3 LA TORTUGA EN EL LOGO

Para poder empezar en alguna parte hay que hacerlo con una herramienta, y en el LOGO se hace con la TORTUGA que en este caso es la herramienta de este lenguaje. Asi pues comenzaremos con la TORTUGA que es un animalito cibernetico controlado por un Computador y vive en la pantalla del mismo.

Las ordenes que se dan a la TORTUGA se efectuan por el teclado de su Computadora diciendola que por ejemplo se mueva hacia adelante 100 unidades (fd 100) y el animalito avanza por el suelo de su pantalla las 100 unidades ordenadas. Si se desea que gire a la derecha 90 grados, bastara conque se lo ordenemos (rt 90) y la TORTUGA girara sus 90 grados que nosotros deseamos, y asi sucesivamente, por lo que la persona, tanto infantil como adulta, ira introduciendo estas ordenes y creara un dibujo interesandose en el mundo de la informatica pudiendo llegar a inventar nuevos comandos.

12.4 INTRODUCCION AL LOGO

Un vez que Vd. tenga en su pantalla el signo de control (?), podra introducir por el teclado las ordenes pertinentes, con lo cual empezara tecleando con letras minusculas:

```
fd 50 <cr>
```

viendo que le aparece la Tortuga moviendose hacia adelante 50 pasos de pantalla y dibujando una linea tras ella, desde donde empezo hasta donde termino las 50 unidades.

Con este paso, podemos deducir que la orden "fd" significa hacia adelante (FORWARD en ingles). El siguiente paso sera el teclear la orden (siempre en minusculas):

```
rt 90 <cr>
```

viendo que la Tortuga gira 90 grados a la derecha, deduciendo que la orden "rt" significa, giro a la derecha (RIGHT en ingles).

Como comprobara, hasta ahora todo esto es muy facil, por lo que seguiremos con nuestro dibujo introduciendo otra vez:

```
fd 50 <cr>

y

rt 90 <cr>
```

teniendo en la pantalla un cuadrado perfecto, con lo cual Vd. podra probar el introducir todas las ordenes de una vez y ver los resultados:

```
fd 50 rt 90 fd 50 rt 90 <cr>
```

SORPRESA!!! el dibujo que aparece es exactamente igual al anterior por lo que deducimos que las ordenes se pueden introducir todas a la vez obteniendo unos resultados sorprendentes.

Una vez que Vd. haya practicado con estas dos ordenes del lenguaje LOGO, podremos proseguir con nuestra esplicacion de este sorprendente lenguaje.

12.5 PROCEDIMIENTOS DE LAS ORDENES

Hay una serie de procedimientos en este lenguaje que los podremos denominar como "sencillos o primitivos" los cuales se pueden difurcar para hacer sus propias tareas.

Entre los procedimientos sencillos, podemos incluir los conocidos hata este momento "fd" y "rt" que fueron comentados anteriormente. A estos dos anadiremos los de "bk" que significa hacia atras (BACK en ingles), "lt" que se interpreta como giro hacia la izquierda (LEFT en ingles) y por ultimo "cs" que su mision es el borrar todo lo que hay en la pantalla (CLEAR SCREEN).

Como Vd. comprobo anteriormente, se repitio los procedimientos cuatro veces para reproducir un cuadrado, pues bien, esto se puede conseguir con un procedimiento muy sencillo el cual lo vera a continuacion. Introduzca lo siguiente:

```
cs <cr>
repeat 4 [fd 50 rt 90] <cr>
```

El resultado sera que primero se borra la pantalla y luego una vez introducido el segundo procedimiento, le dibujara un cuadrado perfecto identico al que anteriormente conseguimos.

Otra de las ventajas de este lenguaje, es que se le puede indicar que efectue una tarea simplemente dandola un nombre, el cual almacenara el procedimiento y simplemente con pedir el nombre de la tarea lo ejecutara, asi pues para que Vd. vea su ejecucion, teclee lo siguiente:

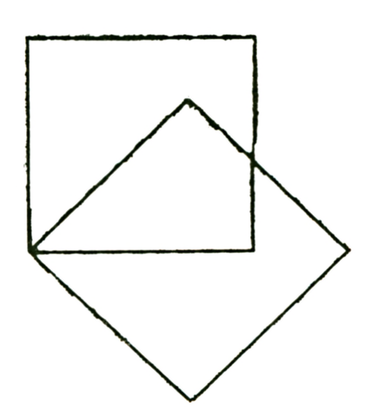
```
cs <cr>

to cuadrado <cr>
repeat 4 [fd 50 rt 90] <cr>
end <cr>
```

Esto significara, que cuando el LOGO encuentre la palabra "cuadrado" interpretara el procedimiento dibujando un cuadrado perfecto, asi pues practicaremos con otra variante tecleando lo siguiente:

```
cs <cr>
cuadrado rt 45 cuadrado <cr>
```

consiguiendo dos cuadrados desplazados el uno del otro en 45 grados y con 50 unidades en cada lado como muestra el dibujo.



12.6 LOS PARAMETROS EN LOS PROCEDIMIENTOS

La posibilidad de hacer un procedimiento como la de una tarea mediante un procedimiento que le hemos dado anteriormente, es una de las ventajas del Dr. LOGO, llamado tambien razonamiento variable.

Para poder trazar unos cuadrados con diferentes tamanos de lado, modificaremos la definicion de cuadrado de la siguiente manera:

```
cs <cr>

to cuadrado2 :lado <cr>
repeat 4 [fd :lado rt 90] <cr>
end <cr>
```

Con este nuevo concepto, hemos introducido una variable, que en este caso la deniminamos :lado, la cual siempre tiene que ser precedida por el caracter de dos puntos (:), no siendo esta variable un directivo para ejecutar alguna accion. Lo siguiente que notamos es que el comando "fd" normalmente va seguido por un numero que puede ser variable indicando el numero de unidades hacia adelante. En el "cuadrado2" tenemos un argumento, que en este caso es ":lado" representando todos los distintos valores que se pueden dar al lado del cuadrado.

Otra variante para poder dibujar un cuadrado, seria de esta sencilla forma:

```
cuadrado2 100 <cr>
```

obteniendo un dibujo de un cuadrado de 100 unidades de lado, pero si introducimos lo siguiente:

```
cuadrado2 100 cuadrado2 200 <cr>
```

obtendremos el siguiente dibujo:

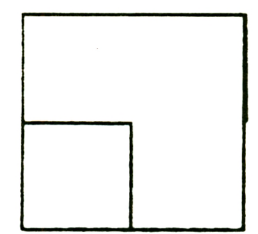


Figura 12.2

produciendose **dos cuadrados con un vertice comun** pero un el doble de tamano que el otro, **pero si variamos la ultima** linea de comando de la siguiente forma:

```
cuadrado2 100 lt 45 cuadrado2 200 <cr>
```

el resultado **sera el siguiente:**

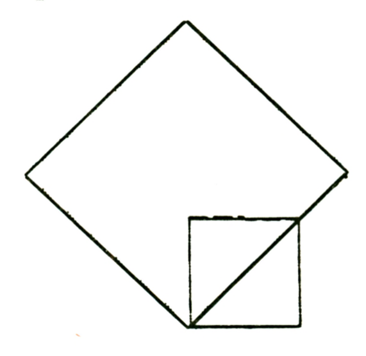


Figura 12.3

Hay infinidad de variantes en este tipo de operaciones, habiendo una que nos permite usar unas variables para memorizar valores, ademas de poder transferir dichos valores a un procedimiento. Para ello se definira un procedimiento que se encarge de dibujar un triangulo, asi pues se escribira:

```
cs <cr>
to triangulo <cr>
repeat 3 [fd :lado rt 100] <cr>
end <cr>
```

donde en este procedimiento, el :lado (lado del triangulo) es la variable, y para poderlo comprobar se tecleara lo siguiente:

```
make "lado 75  <cr>
triangulo <cr>
```

Este procedimiento lo que hace es que al "lado" le da un valor de 75 y a continuacion le indicamos que dibuje un triangulo, pudiendole pedir que nos de el valor almacenado tecleando solamente ":lado" inmediatamente despues del signo * dandonos el valor de esa variable.

Por ultimo, se puede usar esta variable para cualquier otro procedimiento, como dibujar un determinado "patron" por lo que para demostrarlo teclearemos lo siguiente:

```
to patron <cr>
triangulo lt 50 triangulo rt 50 <cr>
make "lado :lado + 4 <cr>
patron <cr>
end <cr>
make "lado 10 <cr>
cs patron <cr>
```

Se podra observar que el valor de :lado se va incrementando una cantidad a su valor previo, asi pues cada vez que se dibuja un "patron" lo hace a un tamano mucho mayor pudiendose parar cuando uno quiera pulsando solamente la tecla "ESC".

CAPITULO 13

13.1 EDITANDO CON LOGO

En el lenguaje del Dr. LOGO tenemos la posibilidad de poder corregir los errores que se cometen al introducir los procedimientos o cualquier escritura atraves de nuestro teclado.

Para poder accionar la edicion hay una serie de teclas fundamentales que son necesarias para esta operacion, las cuales son las siguientes:

En el caso de que se desee Editar un procedimiento existente, bastara con teclear el comando "ed" (significativo de editor pero sin las comillas) mostrando el Dr. LOGO la ultima version del procedimiento, pudiendose usar los comandos que detallamos a continuacion para efectuar las correcciones pertinentes.

Las cuatro teclas de movimiento de cursor (↑), (↓), (⟶), y (⟵) situadas en la parte superior derecha del teclado, sirven para el desplazamiento del cursor una vez por cada pulsacion de la tecla. En el caso de necesitar el desplazamiento del cursor una pagina entera tanto hacia arriba como hacia abajo, y una linea entera cada vez a la izquierda o a la derecha, de deberan pulsar una de las teclas de movimiento del cursor manteniendo pulsada la tecla CTRL.

En el caso de necesitar borrar un caracter del procedimiento, se podra usar la tecla CLR si el cursor esta exactamente en el caracter a borrar, y en el caso de que el cursor este en una posicion a la derecha de caracter, se podra usar la tecla DEL para borrar dicho caracter.

La tecla de ENTER se usa para indicar al Computador que hemos terminado con la revision de la linea.

En el caso de querer abandonar la operacion de Edicion, bastara con pulsar la tecla de ESC, y si se quiere terminar de editar un procedimiento, tendra que pulsar la tecla COPY.

Como Vd. puede ver, los comandos que se usan en esta operacion, son muy secillos pero muy utiles a la hora de tener que hacer correcciones, por lo que le sugerimos que practique un poco con todos ellos para poder ver su gran utilidad.

13.2 CREACION DE UNA LISTA

En el momento en que Vd. este cansado de jugar con los graficos, el Dr. LOGO le ofrece una forma de utilizar los procedimientos, contando con un proceso de listas y funciones tomadas de un lenguaje de Inteligencia Artificial.

Una lista en el lenguage LOGO es algo que normalmente se encuentra encerrado entre corchetes ([]) teniendo metodos convencionales de presentar la lista de instrucciones que sigue a un comando REPEAT. Asi pues, en la linea:

```
repeat 1 [fd 100 lt 37 bk 100]
```

el comando REPEAT espera una lista a continuacion del primer parametro, que en este caso es 1, siendo la lista el segundo parametro.

Hay un comando denominado RUN (ejecucion) cuyo parametro es una lista, y cuya funcion es el tratar dicha lista como si fuese una secuencia de comandos que se han introducido por el teclado. Por lo tanto, la secuencia:

```
run [fd 100 lt 37 bk 100]
```

indicara que la palabra RUN hara las mismas funciones que REPEAT 1 del ejemplo anterior.

Otro ejemplo de como se podria usar el comando RUN seria el siguiente:

```
to accion.variable :instrucciones
run instrucciones
end
```

Asi pues:

```
accion.variable [repeat 4[fd 50 rt 90]]
```

dibujara un cuadrado de 50 unidades, mientras que:

```
accion.variable [fd 100]
```

solamente dibujara una linea recta de cien unidades de pantalla.

Los graficos de TORTUGA por si solos son tan utiles, que una version de los graficos de LOGO constituye una herramienta inmejorable para la ensenanza y el aprendizaje.

13.3 EL LENGUAJE INMEDIATO

En el lenguaje LOGO se ejecutan las instrucciones cuando se introducen por el teclado, diferente a otros lenguajes, como el BASIC que trabaja cuando se ejecuta el programa mediante el comando RUN. En otros lenguajes, como PASCAL, las instrucciones del programa tienen primero que almacenarse, compilarse, y solo entonces puede ejecutarse el programa siendo este metodo el mas pesado, asi como el LOGO el mas inmediato.

13.3.1 Una secuencia

Como hemos mencionado anteriormente, el lenguaje LOGO tiene la virtud de poderse ejecutar de inmediato, por lo que si Vd. introduce el comando:

```
fd 100
```

vera que se dibuja una linea recta de 100 unidades tan pronto haya pulsado la tecla de ENTER, debiendo terminar todas las lineas de comandos con un retorno de carro <cr> (ENTER) dado que si no el Computador no sabria cuando se ha llegado al final de la linea.

Por otro lado, si Vd. introduce el siguiente comando:

```
rt 90
```

comprobara que aparentemente no pasa nada, salvo que se puede apreciar que la TORTUGA ha cambiado de direccion de la que estaba apuntando sucediendo todo de una forma inmediata.

La posibilidad de poder dibujar figuras bastante complejas por mediacion del teclado, aunque esto le parezca infantil o inutil, es una forma comoda de ir aprendiendo, llamandolo a esto "Razonamiento Secuencial". Asi pues, si la forma de ejecutar la accion B antes que la C, y para llevar a cabo la ejecucion de la accion B es preciso haber ejecutado antes la accion A, por lo tanto es evidente que la secuencia de acciones debera ser A,B,C.

Consideremos estos tres comandos:

```
fd 50
rt 90
fd 100
```

el resultado sera el de esta figura:

Figura 13.1

A continuacion efectuaremos un pequeno cambio alterando el orden:

```
fd 50
fd 100
rt 90
```

Teniendo como resultado el de esta figura:

Figura 13.2

Como comprobara el resultado de la figura es totalmente diferente al anterior, apareciendo solamente una linea recta apuntando hacia arriba dado que esa es la forma en que se ha supuesto que apunta la TORTUGA.

El tercero y ultimo cambio, sera el siguiente:

```
rt 90
fd 50
fd 100
```

cuyo resultado seria una linea recta como esta:

Figura 13.3

13.3.2 El LOGO como calculadora

Hasta ahora hemos podido dibujar una figura con la TORTUGA, pero tambien se puede usar el Dr. LOGO como una calculadora instantanea, con lo cual Vd. podra efectuar un sin fin de operaciones.

Para demostrar la accion de este modo de calculo, introduciremos la siguiente operacion a modo de demostracion:

```
print 11-23/5
```

se obtendra una salida en pantalla de:

```
>6.4
```

dado que el Computador actua secuencialmente, encontrando primero el 11, tomando el 23 dividiendolo entre 5, y restando el resultado de la division por 11. El orden en estas operaciones es muy importante, y para demostrarlo entraremos los siguientes ejemplos:

```
print 23-11/5

>20.8
```

pero si lo ponemos de la siguiente forma:

```
print (23-11)/5

>2.4
```

Para entender estos ejemplos aritmeticos tan simples, se necesitara saber en que orden ocurren sus acciones, asi pues si introducimos una suma como:

```
2+5
```

el LOGO nos dara un error dado que el sistema no sabe que hacer con el siete (7) que ha resultado de la suma.

Otra forma de introducir una funcion aritmetica, seria la siguiente:

```
fd (2+5)

o

lt (2+5)
```

pero seria necesario el ordenarle que hiciese algo dado que no tiene ningun sitio donde poner el 7 resultante, asi pues a continuacion pondremos una serie de comandos donde si nos dara un resultado logico:

```
fd 30 rt (0+10)
fd 30 rt (10+10)
fd 30 rt (20+10)
fd 30 rt (30+10)
```

```
fd 30 rt (40+10)
fd 30 rt (50+10)
```

y asi sucesivamente hasta que Vd. decida parar, con lo cual se producira unas cuantas lineas y giros.

13.3.3 El LOGO como lista de elementos

Como hemos mencionado anteriormente, el comando REPEAT y el numero de repeticiones se denomina LISTA, esperando el LOGO a que le ordenemos ejecutar algo, por lo que pondremos este ejemplo como LISTA:

```
[Esto es una prueba]
```

Al ejecutar este ejemplo se producira un error dado que es inprescindible que informemos al sistema que debe hacer con esta LISTA, y dado que lo mejor que podemos hacer con esta LISTA seria el sacarla por pantalla para poderla visualizar, escribiremos lo siguiente:

```
print [Esto es una prueba]
>Esto es una prueba
```

pero si escribimos:

```
print first [Esto es una prueba]
```

lo que se obtendra en la pantalla seria:

```
>Esto
```

Por lo tanto aqui tenemos un nuevo procedimiento el cual es FIRST (primero) el cual extrae de la LISTA su primer elemento.

El procedimiento inverso a FIRST es "bf" (BUTFIRST que significa sin el primero) que lo que hace es extraer la lista escepto su primer elemento, y para demostrar las diversas variantes introduciremos lo siguiente:

```
print bf [Esto es una prueba]
>es una prueba
```

pero si lo variamos a:

```
print first bf [Esto es una prueba]
>Es
```

sin embargo si ponemos:

```
print bf first [Esto es una prueba]
>sto
```

Habra visto las diferentes variaciones que se pueden hacer con un procedimiento, y lo educativo que es este lenquaje por sus diversas aplicaciones dado la versatilidad del mismo.

Para ampliar lo anteriormente espuesto, mencionaremos que el procedimiento FIRST puede actuar sobre una palabra, por lo que para esplicarlo escribiremos:

```
print first "EJEMPLO
>E
```

Aqui las comillas (") que preceden a la palabra EJEMPLO indican que nos referimos a esa palabra como tal, y no a su valor o su accion como procedimiento.

Para esplicar todo esto con sus variantes le diremos que:

1.- EJEMPLO es tratado por el Dr. LOGO como un procedimiento
2.- "EJEMPLO representa el nombre de algo
3.- :EJEMPLO se interpreta como el valor de una cosa llamada EJEMPLO

Aclararemos que en estos Capitulos referentes al leguaje LOGO no hemos detallado la serie de comandos que este tiene, dado que en su manual de usuario viene detallado con mucha precision, y seria una duplicidad de lectura.



14.1 ALGUNOS EJEMPLOS DE PROGRAMAS EN LOGO

En este Capitulo pondremos algunos ejemplos para demostrar como la TORTUGA le ayuda a dibujar con un simple programa y entender como puede combinar procedimientos con palabras para crear programas importantes.

Empezaremos explicando un programa que le dibuja en pantalla el perfil de una casa. Este programa es bastante sencillo y se comenzara por el procedimiento final para luego desglosar los restantes procedimientos.

El programa principal es el siguiente:

```
TO CASA
CUADRADO
FD 30
RT 30
TECHO
LT 30
BK 30
RT 90
FD 15
LT 90
PUERTA
RT 180
HT
END
```

En este programa, lo primero que se le indica a la TORTUGA es que dibuje un CUADRADO cuyo procedimiento es el siguiente:

```
TO CUADRADO
REPEAT 4 [FD 30 RT 90]
END
```

Dado que la TORTUGA se ha parado en la esquina inferior izquierda, tendremos que desplazarla a la esquina superior izquierda, asi pues hemos introducido el comando FD 30 posicionandola en su sitio. A continuacion se debera rotar la TORTUGA para poder emplazar el techo correspondiente, y para ello hemos puesto el comando RT 30.

Dado que en este momento la TORTUGA la tenemos posicionada para seguir el dibujo, introduciremos el procedimiento del TECHO, el cual es el siguiente:

```
TO TECHO
REPEAT 3 [FD 30 RT 120]
END
```

Una vez dibujado el TECHO necesitaremos una serie de comandos para posicionarnos en el lugar donde tendremos que dibujar la puerta, los cuales son los siguientes: LT 30, BK 30, RT 90, FD 15 y LT 90. Una vez posicionada la TORTUGA podremos proseguir con el dibujo, asi pues llamaremos al procedimiento PUERTA el cual es el siguiente:

```
TO PUERTA
FD 12
RT 90
FD 7
RT 90
FD 12
END
```

En este paso, tenemos dibujada la PUERTA de la casa, pero como Vd. notara, la TORTUGA esta posicionada en el dibujo lo cual afea el mismo, por lo tanto sera necesario el introducir los comandos RT 180 y HT (HideTurtle) para que desaparezca la TORTUGA asi terminar el dibujo.

Dado que tenemos varios procedimientos en memoria, haremos uso de ellos para jugar un poco creando nuevos dibujos e introducir comandos nuevos demostrando la versatilidad de este lenguaje.

Para empezar, usaremos el procedimiento TECHO e implementaremos un nuevo comando llamado "SETPC" (que significa SET Pen Color) para que la TORTUGA haga su dibujo en un color especifico, asi pues escribiremos el siguiente programa:

```
?setpc 3
?repeat 3 [techo fd 20 lt 120]
```

El resultado de este programa sera que apareceran en pantalla tres triangulos los cuales corresponden al procedimiento del TECHO.

Otra variante que se puede usar con este lenguaje sera el usar dos procedimientos para crear un dibujo, con lo cual en este caso usaremos los de TECHO y CUADRADO que todavia los tenemos en memoria. Asi pues para que Vd. pueda ver este interesante dibujo en su pantalla, escribiremos el siguiente programa:

```
?setpc 4
?repeat 6 [techo cuadrado lt 60]
```

Complicaremos un poco mas la cosa, pero siempre usando los procedimientos que tenemos almacenados en memoria, y en este caso usaremos, ademas del de CUADRADO, el comando REPEAT dentro de un procedimiento para obtener un fantastico dibujo, por lo tanto escriba Vd. lo siguiente:

```
?setpc 1
?repeat 8 [repeat [fd 40 rt 60] cuadrado lt 45]
```

Otra combinacion con los procedimientos de TECHO PUERTA seria el siguiente:

```
?repeat 6 [techo fd 30 puerta rt 120]
```

Pero si a este ejemplo le cambiamos el comando rt (right) por el de lt (left), el resultado seria:

```
?repeat 6 [techo fd 30 puerta lt 120]
```

comprobando la diferencia de dibujo que puede hacer un cambio de un comando dentro de un programa.

Como vera a continuacion, el leguaje LOGO hace muchas mas cosas que dibujar con nuestra amiga la TORTUGA, dado que se pueden combinar numeros y palabras, asi pues una demostracion de como este lenguaje hace un tratamiento con numeros, seria el siguiente:

```
?print 34 + 46 + 57
>137
?print 157.50 - 34.25
>123.25
```

Estos dos ejemplos, como Vd. puede comprobar, calculan una suma y una resta, pero tambien puede hacerse multiplicaciones y divisiones. En el caso de las multiplicaciones, el signo que se emplea es el de "*" y no el de la "X", asi pues:

```
?print 25 * 15
>345
?print 15 / 16
>.9375
```

Ademas de estas cuatro operaciones fundamentales, el Dr. LOGO tiene la capacidad de resolver problemas de "raices cuadradas, senos, cosenos, etc".

14.2 DIBUJOS GEOMETRICOS

Generalmente la TORTUGA aparece en la pantalla en el centro de esta y apuntando hacia arriba, por lo que las coordenadas del centro son X 0 (eje horinzontal) y Y 0 (eje vertical), midiendose los angulos en el sentido de las agujas de un reloj.

Uno de los dibujos geometricos de demostracion es el denominado "Arbol" el cual le ira demostrando la contruccion de un dibujo teniendo como base un angulo de 90 grados.

Este tipo de dibujo tiene un origen recursivo, y su procedimiento para producirlo es el siguiente:

```
to arbol :longitud :orden
if :orden = 0 then [stop]
lt 45
fd :longitud
arbol :longitud/2 :orden-1
penup
back :longitud
rt 90
pendown
fd :longitud
arbol :longitud/2 :orden-1
penup
back :longitud
lt 45
pendown
end
```

Ignoraremos por el momento la importancia de :ORDEN y las llamadas recursivas a ARBOL, asi pues lo primero que hace la TORTUGA es girar 45 grados a la izquierda dado que originalmente estaba apuntando hacia arriba, por lo tanto el giro de 45 grados es respecto a la vertical. A continuacion se le ordena que avance :LONGITUD de unidades, teniendo dibujada una linea a la izquierda. El siguiente comando (penup) le indica que la pluma no escriba para poder volver hacia atras sin dibujar pero manteniendonos en la misma direccion, una distancia de :LONGITUD estando en este momento donde empezamos.

Dado que tenemos la TORTUGA mirando a 45 grados a la izquierda y lo que queremos es que este mirando a 45 grados a la derecha con respecto a la vertical, la tendremos que hacer girar 90 grados ademas de tener que bajar la pluma para que pueda seguir escribiendo, la orden de avance de :LONGITUD de unidades con lo que una vez se ha dibujado esta linea la pluma sube otra vez regresando la TORTUGA al lugar de partida, haciendo un giro de 45 grados a la izquierda para dejarla como estaba antes de empezar a dibujar y bajando la pluma.

A continuacion veremos el segundo parametro :ORDEN, que si su valor es 0 el procedimiento se detendra, y en caso contrario dibujara una linea inclinada a 45 grados respecto a la vertical, activandose el procedimiento ARBOL una vez terminada. Activado el procedimiento ARBOL, esta vez con la longitud de las lineas dividida por dos, y la orden reducida en 1 producira el siguiente dibujo simetrico, y asi sucesivamente siempre que la :ORDEN-1 sea distinta a cero.

En el caso de que se incremente el parametro :ORDEN, aumentara la complejidad del dibujo, asi pues si el valor de ORDEN se aumenta en 10 y el valor final de :LONGITUD despues de 10 divisiones por dos, es aproximadamente mil veces mas pequeno que el valor inicial, por lo que los incrementos seran tan pequenos que llegaran a no poderse dibujar.

Las posibilidades de diferentes parametros para hacer dibujos complejos son infinitos, por lo que debera practicar para poder ser un especialista en este tipo de lenguaje.

CAPITULO 15

15.1 SISTEMA OPERATIVO AMSDOS

La mision de todo Sistema Operativo en un Computador esta destinada a la comunicacion con todos los dispositivos, asi como mantener el control de los mismos para su comunicacion entre ellos.

El sistema operatico AMSDOS esta diseñado para los equipos AMSTRAD permitiendo que los programas de BASIC tengan acceso a los ficheros de disco de una forma similar a los ficheros de cassette. Esta ventaja del AMSDOS nos permite igualmente, el que los programas en cassette puedan manejar ficheros en disco con minimas modificaciones con la unica incopatibilidad en los nombres de dichos ficheros, dado que para los ficheros en disco deberan cumplir las normas de CP/M.

Una de las ventajas de este sistema operativo, es que su estructura a sido disenada para poder complementar al sistema operativo CP/M dado que ambos comparten la misma estructura de los ficheros pudiendo efectuar lecturas y escrituras.

Desde el disco se podran conmutar los cauces de entrada y salida de la cassette, asi pues todas las ventajas que estan disponibles en cinta estaran disponibles en disco, teniendo la posibilidad de poder llamar al directorio del disco, asi como borrar, cambiar nombres de ficheros y poder acceder a la unidad de disco deseada.

15.2 EL DIRECTORIO DEL DISCO Y SU ESTRUCTURA

Dentro de la estructura de un disco, hay dos secciones, una que se dedica al directorio, y la otra que es la zona de los datos. El Directorio almacena todos los nombres de los ficheros, asi como la organizacion del area de los datos, pudiendose calcular la extension de un fichero asi como el espacio libre del disco, el cual se efectua sumando la longitud de todos los ficheros y restandola de la capacidad total del disco.

Una constumbre generalizada es que cuando se va a leer un fichero en disco para ser volcado en memoria, es necesario el consultar el Directorio para ver si esta en el, asi mismo es necesario consultar el Directorio, cuando se piensa escribir en el, para ver si se tiene espacio suficiente en el mismo para hacer la reserva de espacio correspondiente.

La estructura de los ficheros en AMSDOS esta primordialmente representada por el nombre que se le da al fichero para su futura identificacion, constando dicho nombre de dos partes separadas por un punto (.), siendo la primera una palabra de ocho caracteres como

maximo, donde se representa el nombre de dicho fichero que siempre tiene alguna relacion con el nombre del programa. La segunda parte puede contener tres caracteres los cuales representan al tipo de programa, asi por ejemplo ASM.COM, PRUEBA.TXT y PROGRA.BAS serian nombres e ficheros.
Los nombres y los tipos pueden consistir de una mezcla de letras y numeros pero no pueden tener espacios entre ellos. A continuacion le detallamos algunos tipos que se usan abitualmente:

. <espacios>	En este caso no se ha especificado el tipo, pudiendo ser un fichero que se ha creado con OPENOUT, o tambien un fichero de un programa en BASIC salvado con SAVE "nombre",P
.COM	Este tipo de fichero es usado en CP/M como fichero de orden.
.BIN	Programa grabado con SAVE "nombre",B (en binario)
.BAS	Este es un tipo de programa en BASIC que se ha podido grabar con SAVE "nombre", o SAVE "nombre",P o bien con SAVE "nombre.BAS",A
.TXT	Todo programa que normalmente tiene este tipo, es un fichero que contiene texto, como por ejemplo de un procesador de palabra.

En el caso de que Vd. tenga instalado en su Computador dos unidades de Disco, estas unidades seran denominadas com A y B las cuales deberan ser asignadas dado que el sistema no hace una busqueda de su fichero por las dos unidades. Para asignar esta unidades se podran utilizar las ordenes ¦A o ¦B, pudiendose tambien usar el :A o :B sin tener que modificar la unidad implicita. Asi por ejemplo se podra poner:

```
¦B
SAVE "PROGRAMA.BAS"
¦A
```

o bien

```
¦A
SAVE "B:PROGRAMA.BAS"
```

En los dos ejemplos el PROGRAMA.BAS sera grabado en la unidad de Disco B.

15.3 ORDENES DEL SISTEMA OPERATIVO AMSDOS

Las ordenes externas del sistema operativo AMSDOS son las siguientes:

¦A

Indica que la unidad principal de control es la unidad B

¦B

Indica que la unidadd principal de control es la unidad B

¦CPM

Carga en el Computador el sistema operativo CP/M 2.2

|DIR

Le saca por pantalla el directorio del disco y su espacio libre de uso, asi por ejemplo se podra escribir: |DIR,"*.BAS"

|DISC

Es equivalente a las ordenes |DISK.IN y |DISC.OUT

|DISC.IN

Esta orden indica que todas las operaciones de lectura se deberan hacer con el disco.

|DISC.OUT

Esta orden indica que todas las operaciones de escritura se deberan hacer con el disco.

|DRIVE

Este tipo de orden establece la unidad de disco principal, asi por ejemplo: |DRIVE,"A" establecera que la unidad principal es la A.

|ERA

Esta orden borrara todos los ficheros que en la orden esten entre comillas ("), como por ejemplo: |ERA,"*.BAS"

|REN

Es una orden que permite cambiar de nombre un fichero existente en disco siempre que no este en el disco el nuevo nombre, como por ejemplo: |REN,"nuevo.COM","viejo.COM"

|TAPE

Esta orden se usa con una unidad de Cassette externa, y equivale a las ordenes |TAPE.IN y |TAPE.OUT

|TAPE.IN

El uso de esta orden indicara que solamente se haran lecturas desde la Cassette que esta conectada al Computador exteriormente, o interiormente.

|TAPE.OUT

El uso de esta orden indicara que solamente se haran escrituras desde la Cassette que esta conectada al Computador exteriormente, o interiormente.

En el Apendice A le mostraremos los diferentes mensajes de errores que apareceran bajo el Sistema Operativo AMSDOS.

APENDICE A

MENSAJES DE ERROR DEL AMSDOS

Bad command

Este mensaje indica que el comando que se ha entrado es incorrecto por error sintactico, o por que no lo reconoce el Amsdos.

<Nombre de fichero> already exist

La indicacion de este mensaje significa que se ha intentado cambiar el nombre de un fichero que ya existe en el directorio del disco.

<Nombre de fichero> not found

Si se intenta acceder, borrar o cambiar un fichero que no esta en el disco, saldra este tipo de mensaje.

Drive d:directory full

Este mensaje indica al usuario que el directorio esta lleno.

Drive d:disc full

La significacion de este mensaje indica que el disquete que esta en la unidad d: esta lleno dado que no hay mas bloques disponibles, teniendo que cambiar el disquete por otro formateado.

Drive d:disc changed, closing <nombre de fichero>

Este mensaje de error indica al usuario que en la unidad de disco d: se ha cambiado el disquete, y el sistema esta cerrado al fichero con ese nombre.

<Nombre de fichero> is read only

Cuando aparece este tipo de mensaje, es por que se esta intentando borrar o cambiar el nombre de dicho fichero que esta protejido para escritura.

MENSAJES DE ERROR DEL BIOS

Todos los mensajes de error del Bios, van precedidos de la siguiente pregunta:

Retry, Ignore or Cancel?

queriendo significar que: se reentre, se ignore o se cancele la operacion que se intentaba ejecutar. Para este mensaje, se deberan usar las teclas de R, I, o C segun su decision a tomar, desechando el sistema cualquier otra tecla que se pulse.

Drive d:disc missing

Este mensaje aparecera cuando el Bios intenta acceder a la unidad de disco d: y no se encuentra ningun disquete o la trampilla esta mal cerrada.

Failed to load boot sector

En el caso de que el sistema falle en la carga del BOOT en CP/M, saldra este mensaje, asi como tambien puede salir si la lectura es incorrecta o por que los bytes de ese sector tengan el mismo valor.

Failed to load CP/M

Este mensaje saldra durante la carga en frio, dado que un sector del CCP o del BDOS no se ha leido correctamente.

Drive d:disc is write protected

Cuando se esta intentando escribir en un disco que este protejido en escritura, aparecera este mensaje dado que el BIOS esta intentando escribir.

Drive d:read fail

La indicacion de este mensaje indica que Vd. ha tenido un error cuando intentaba acceder a leer unos datos del disco.

Drive d:write fail

Al igual que el anterior error, este mensaje le indica que el sistema ha detectado un error en escritura cuando Vd. intentaba escribir algun dato en el disco.

APENDICE B

ORDENES EN CP/M

En este apendice, se resumen cada una de las ordenes permanentes y transitorias del CP/M, listando dichas ordenes en orden alfabetico.

Ordenes ASM

ASM nombre de fichero <cr>

Esta orden ensamblara el fichero "nombre.ASM", usando el disco actualmente catalogado.

ASM nombre de fichero.opt <cr>

Esta orden ensamblara el fichero "nombre.ASM" en la unidad o:(A:,B:) escribiendo un fichero del tipo HEX en la unidad p:(A:,B:), o salta si p: es Z. Ademas escribe un fichero con el tipo PNR en la unidad t:(A:,B:) y lo saca por consola si p: es X.

Ordenes DDT

DDT <cr>

Esta orden cargara en memoria el programa DDT.COM

DDT d:nombre de fichero.tipo <cr>

Cuando se introduce esta orden, se cargara en memoria el programa DDT.COM asi como el fichero "nombre.tipo" de la unidad d: para que se le pueda examinar, modificar o ejecutarlo.

Assss

Esta orden introducira las sentencias en lenguaje ensamblador, comenzando en la direccion de memoria ssss hexadecimal.

D

Le sacara por pantalla el contenido de los 192 octetos de memoria siguientes.

Dssss,xxxx

Le sacara por pantalla el contenido de la memoria desde la direccion ssss hasta la direccion xxxx inclusives.

Fssss,xxxx,cc

Esta orden rellenara la memoria desde la direccion ssss hasta la xxxx con la constante hexadecimal de ocho bits cc.

G

Esta orden empezara la ejecucion del programa a partir de la direccion contenida en el contador del programa.

G,aaaa

Para colocar un punto de ruptura de secuencia en la direccion aaaa hexadecimal, se utilizara esta orden y asi se comenzara la ejecucion en la direccion contenida en el contador del programa.

G,aaaa,bbbb

Como en el caso anterior, esta orden coloca puntos de ruptura en las direcciones hexadecimales aaaa y bbbb y comienza la ejecucion en la direccion contenida en el contador del programa.

Gxxxx

Esta orden hace comenzar la ejecucion en la direccion hexadecimal xxxx.

Gxxxx,yyyy

Esta orden coloca un punto de ruptura en la direccion yyyy y comienza la ejecucion en la direccion xxxx.

Hx,y

Suma y resta de x e y en hexadecimal

Inombre de fichero.tipo

Esta orden establece el bloque de control de fichero por defecto, usando el fichero "nombre.tipo".

L

Hace un listado de las once lineas siguientes de un programa en lenguaje ensamblador.

Lxxxx

Esta orden lista las once lineas siguientes del programa ensamblador en memoria, comenzando en la direccion xxxx.

Lxxxx,yyyy

Esta orden listara las once lineas siguientes del programa en lenguaje ensamblador en memoria, comenzando en la direccion xxxx y terminando en la yyyy.

Mxxxx,yyyy,zzzz

Esta orden traslada el contenido del bloque de memoria que comienza en la direccion xxxx y termina en la yyyy al bloque de memoria que comienza en la direccion zzzz.

R

Para leer un fichero del disco y volcarlo en memc-ia se debera usar esta orden, pero usando antes la orden I.

Rnnnn

Esta orden lee un fichero del disco y lo vuelca en memoria comenzando en la direccion nnnn mas alta que la normal.

Sxxxx

En el caso de tener que visualizar el contenido de la celda de memoria en la direccion hexadecimal xxxx, tendra que usar esta orden, y opcinalmente cambiar su contenido.

Txxxx

Esta orden trazara la ejecucion de la intruccion xxxx del programa.

Uxxxx

Esta orden ejecutara las instrucciones xxxx del programa parando a continuacion y visualizandolo en pantalla el contenido de los registros de la CPU.

Wnombre de fichero.tipo,xxxx,yyyy

Para escribir en el fichero "nombre.tipo" el contenido de la memoria desde xxxx hasta yyyy se debera usar esta orden.

X

Esta orden visualiza en pantalla los registros de la CPU.

Ordenes de DIR

DIR d: <cr>

Saca por pantalla el directorio de todos los ficheros de la unidad de disco d: (A: o B:).

DIR d:nombre de fichero.tipo <cr>

Muestra en pantalla el directorio de todos los ficheros de la unidad d: (A: o B:) donde los nombres concuerden con el ncmbre del fichero.

Ordenes DUMP

DUMP d:nombre de fichero.tipo <cr>

Muestra en pantalla las equivalencias hexadecimales de cada octeto almacenado en el fichero "nombre.tipo" de la unidac d; (A: o B:).

Ordenes ED

ED d:nombre de fichero.tipo <cr>

Esta orden llamara al programa ED.COM (editor) buscando sitio en la unidad de disco d: (A: o B:) para crear un fichero temporal "nombre.$$$" con el texto editado.

nA

Esta orden traslada n numero de lineas desde el fichero original al buffer del editor, si n es cero (ØA) trasladara todas las lineas hasta que el buffer este lleno o hasta la mitad como minimo.

+/-B

Esta orden trasladara el puntero de caracteres al principio o al final del buffer del editor

+/-nC

Esta orden trasladara el puntero de caracteres n posiciones de caracter hacia delante o hacia atras.

+/-nD

Eliminara n caracteres antes o despues del puntero.

E

Esta orden indica que se ha terminado la sesion de edicion cerrando los ficheros y devolviendo el control al CP/M.

nFcadena^Z

Esta orden linkea una cadena.

H

Coloca el puntero de caracteres a la cabecera del fichero editado.

I <cr>

La funcion de esta orden es que todo lo que se escriba por el teclado se introducira en el buffer del editor a partir del puntero de caracteres.

Icadena^Z

Esta orden es identica a la anterior, con la diferencia que esta inserta una cadena.

+/-nK

Elimina n numero de lineas hacia adelánte o hacia atras a partir del puntero de caracteres.

+/-nL

Esta orden traslada el puntero de caracteres al comienzo de la linea donde esta y a continuacion lo traslada n numero de lineas hacia delante o hacia atras.

O

Esta orden lo que hace es vaciar tanto el buffer de edicion como el fichero temporal, regresando al comienzo del fichero original.

+/-nP

Traslada el puntero de caracteres hacia delante o hacia atras, e imprime las paginas.

Q

Esta orden hace retornar el control al CP/M saliendose del Editor, borrando todos los ficheros menos el original que no sufre alteraciones.

R <cr>

Lee el fichero de transferencia en bloque.

Rnombre de fichero <cr>

Esta orden lee un fichero de biblioteca,

+/-nT

Esta orden le mostrara en pantalla n numero de lineas posteriores o anteriores al puntero de caracteres.

ØV

Para poder averiguar el espacio libre en el buffer de edicion, habra de usar esta orden la cual tambien le dara la informacion del numero de octetos libres asi como del tamano total del buffer de edicion.

nW

Esta orden escribira las primeras n numero de lineas del buffer de edicion en el fichero temporal, borrando este numero de lineas de dicho buffer.

nX

Esta orden hara una transferencia del buffer de edicion de las n numero de lineas siguientes al puntero en el fichero temporal X$$$$$$$.LIB anadiendolo al contenido de aquel fichero.

Ordenes ERA

ERA d:nombre de fichero.tipo <cr>

Esta orden borrara el fichero "nombre.tipo" que esta en la unidad de disco d: (A: o B:).

ERA d:*.* <cr>

Esta orden borrara todos los ficheros de la unidad de disco d:

Ordenes LOAD

LOAD d:nombre de fichero <cr>

La funcion de esta orden es la de leer el fichero "nombre.HEX" de la unidad d: (A: o B:) y crear a continuacion un fichero de programa ejecutable "nombre.COM" en la misma unidad de disco.

Ordenes MOVCPM

MOVCPM <cr>

Esta orden preparara una nueva copia de CP/M usando toda la memoria del Ordenador, dando el control de nuevo al CP/M sin salvandolo en disco.

MOVCPM nn <cr>

La funcion de esta orden es identica a la anterior, con la unica diferencia que nada mas nnK numeros de octetos.

MOVCPM ** <cr>

Esta orden es identica a la de MOVCPM con la unica diferencia que la nueva copia es salvada con el SYSGEN o con el SAVE

MOVCPM nn * <cr>

Esta orden es identica a la de MOVCPM nn con la unica diferencia que la nueva copia es salvada con el SYSGEN o con el SAVE.

Ordenes PIP

PIP <cr>

La funcion de esta orden es el cargar en memoria el programa PIP.COM estando preparado para recibir sus ordenes.

PIP linea de orden <cr>

Carga el programa PIP.COM en memoria y ejecuta al mismo tiempo la "linea de orden" retornando al final a CP/M.

d:nuevo.tipo=d:viejo.tipo[p] <cr>

Esta orden copiara el fichero "viejo.tipo" de la unidad d; (A: o B:) en la otra unidad d: (A: o B:) con el nombre "nuevo.tipo" usando el parametro p.

d:nuevo.tipo=d:viejo1.tipo[p],d:viejo2.tipo[q] <cr>

La funcion de esta orden es el crear un fichero con el nombre "nuevo.tipo" en la unidad de disco d:- (A: o B:) cuyo contenido es el del fichero "viejo1.tipo" de la unidad d: usando el parametro p seguido del contenido del fichero "viejo2.tipo" de la unidad d; (A: o B:) usando el parametro q.

d:nombre.tipo=dd:[p] <cr>

Esta orden copiara de la unidad de disco dd: (A: o B:) en el fichero "nombre.tipo" de la unidad d: con el parametro p.

dd:=d:nombre.tipo[p] <cr>

Copiara los datos del fichero "nombre.tipo" de la unidad d: en la unidad de disco dd: (A: o B:) con el parametro p.

dd:=d:[p] <cr>

Esta orden copiara los datos de la unidad de disco d: (A: o B:) en la unidad dd: (A: o B:) con el parametro p.

Parametros PIP

- B Modo de transferencia de bloque.
- Dn Borrara todos los caracteres despues de la columna n
- E Repite la copia tal como se esta produciendo.
- F Elimina los caracteres para el formato de presentacion durante la transferencia.
- Gn Manda al programa PIP que copie un fichero del area n del usuario
- H Verificara el formato del fichero HEX.
- I Ignorar cualquier registro :00 entre formatos HEX.
- L Cambia las letras mayusculas en minusculas.
- N Aumentara un numero de linea a cada una transmitida.
- O Transfiere un fichero objeto.
- Pn Inserta un caracter de definicion de linea despues de cada n linea.
- Qx^Z Manda que se suspenda la copia cuando encuentre la cadena x.
- R Copia un fichero del Sistema.
- Sx^Z Manda que comience la copia cuando encuentre la cadena x.
- Tn Pone un fin de tabulacion en cada n columna.
- U Cambia las letras minusculas en mayusculas.
- V Verifica la copia cuando se ha terminado de hacerse.
- W Copia un fichero R/O.
- Z Pone a cero el bit de paridad.

Ordenes REN

REM nombre1.tipo=nombre2.tipo <cr>

Esta orden cambiara el nombre del fichero "nombre2.tipo" por el de "nombre1.tipo".

Ordenes SAVE

SAVE xxx d:nombre.tipo <cr>

Reserva una parte de memoria de los programas transitorios en el fichero "nombre.tipo" de la unidad de disco d: (A: o B:) en donde la variante xxx es un numero decimal que equivale al numero de paginas de memoria.

Ordenes STAT

STAT <cr>

Esta orden sacara por pantalla todos los atributos asi como el espacio libre de todas las unidades de disco a las que se ha accedido.

STAT d: <cr>

Sacara por pantalla el espacio libre de la unidad de disco d: (A: o B:).

STAT d:nombre.tipo <cr>

Esta orden sacara por pantalla el tamano y los atributos del fichero "nombre.tipo" de la unidad de disco d:.

STAT d:nombre.tipo $atr <cr>

La funcion de esta orden es el asignar el atributo "atr" al fichero "nombre.tipo" de la unidad de disco d:.

STAT DEV: <cr>

Esta orden le informara que unidad esta actualmente esignada.

STAT VAL: <cr>

Le sacara por pantalla las asignaciones de las unidades y un resumen de las lineas de orden STAT.

STAT dl:=df: <cr>

Esta orden asignara la unidad fisica df: a la unidad logica dl:.

STAT USR: <cr>

Mostrara el numero actual de usuario asi como todos los numeros de los ficheros en catalogo del disco.

STAT d:DSK: <cr>

Le indicara las caracteristicas de la unidad de disco d:.

Ordenes SUBMIT

SUBMIT nombre de fichero <cr>

Esta orden creara un fichero $$$.SUB que contendra las ordenes del fichero "nombre.SUB" ejecutando desde ese momento el CP/M las ordenes desde ese fichero y no desde el teclado.

SUBMIT nombre de fichero parametros <cr>

Esta orden creara un fichero $$$.SUB que contendra las ordenes del fichero "nombre.SUB" remplazandose las lineas de orden de este fichero por los parametros durante la creacion del fichero $$$.SUB, ejecutandose desde ese momento el CP/M las ordenes desde ese fichero y no desde el teclado.

Ordene SYSGEN

SYSGEN <cr>

Esta orden cargara en memoria el programa SYSGEN.COM para poder transferir el CP/M de un disquete a otro.

Orden TYPE

TYPE d:nombre de fichero.tipo <cr>

Sacara por pantalla el contenido del fichero "nombre.tipo" que se encuentra en la unidad de disco d: (A: o B:).

Orden USER

USER n <cr>

Esta orden pondra al usuario el numero decimal entero n pudiendose usar desde el Ø al 15 inclusives.

Orden d:

d: <cr>

Pondra como unidad activada la unidad d: (A: o B:).



ORDENES Y COMANDOS EN LOGO

TABLA DE ORDENES LOGO

Instruccion	Contenido	Operador/comando
FS	Seleccion de pantalla grafica	Comando
TS	Seleccion de pantalla de texto	Comando
SS	Seleccion de pantalla partida	Comando
MAKE"palabra	Definicion de una variable	Comando
PRINT objeto"	Muestra el objeto en pantalla	Comando
COUNT objeto	Cuenta elementos	Operador
FIRST objeto	Muestra el primer elemento	Operador
LAST objeto	Muestra el ultimo elemento	Operador
BUTFIRST objeto	Borra el primer elemento	Operador
BUTLAST objeto	Borra el ultimo elemento	Operador
LPUT	Anade al final	Operador
FPUT	Anade al principio	Operador
LIST	Confecciona una lista	Operador
SE	Confecciona una lista de palabras	Operador
WORD	Confecciona palabras	Operador
ASCII palabra	Muestra el codigo ASCII	Operador
CHAR numero	Muestra el caracter ASCII	Operador
TO [variable]	Crea un procedimiento	Comando
END	Pone un fin al procedimiento	Comando
EDIT [nombre]	Activa el editor de procedimientos	Comando
WORD objeto	VERDAD si el objeto es palabra	Operador
LISTP objeto	VERDAD si el objeto es una lista	Operador
NUMBERP objeto	VERDAD si el objeto es un numero	Operador
EMPTY objeto	VERDAD si el objeto esta vacio	Operador
EQUALP obj1 obj2	VERDAD si el obj1 y obj2 son iguales	Operador
MEMBERP obj lista	VERDAD si el obj es un elemento de una lista	Operador
NAMEP palabra	VERDAD si el nombre es variable	Operador
THING objeto	Devuelve el contenido	Operador
EDNS	Activa el editor de variables	Comando
OUTPUT dato	Pone el dato de entrada como dato de salida del procedimiento	Comando
STOP	Para le ejecucion de un subproce-dimiento para retornar al proce-dimiento que lo llamo	Comando
PRINT dato	Saca por pantalla el dato	Comando
TYPE dato	Saca por pantalla el dato sin saltar de linea	Comando
JOY [n]	Indica la direccion en que se desplaza la palanca del joystick	Operador
PADDLE [n]	Indica el giro dado por el boton del paddle	Operador
RL	Da una lista introducida por el teclado	Operador
BEEP	Conecta el zumbador	Comando

TABLA DE ORDENES DE GRAFICOS

Instruccion	Contenido	Operador/comando
FORWARD numero	Hace avanzar la tortuga	Comando
BACK numero	Hace retroceder la tortuga	Comando
RIGHT grados	Giro a la derecha	Comando
LEFT grados	Giro a la izquierda	Comando
HOME	Pone la tortuga en el centro de la pantalla	Comando
CS	Borrado de la pantalla	Comando
PENUP	Levanta la pluma	Comando
PENDOWN	Baja la pluma	Comando
PE	Activa el borrador	Comando
PX	Activa la pluma de color inverso	Comando
PEN	Indica el estado de la pluma	Operador
CLEAR	Borra la pantalla dejando la tortuga en la posicion que ocupaba	Comando
HT	Esconde la tortuga	Comando
ST	Muestra la tortuga	Comando
SHOWNP	VERDAD si la tortuga es visible	Operador
WINDOW	Es activado el modo en que la pantalla solamente representa una parte del campo de la tortuga	Comando
WRAP	Activa el modo en que la tortuga no puede salir de la pantalla	Comando
SETPN n. pluma	Utiliza el numero de la pluma	Comando
SETPC n.pluma color	Utiliza el numero de pluma y color	Comando
SETBG n. color	Pone el fondo del numero de color	Comando
PN	Retorna el numero de la pluma	Operador
PC n. pluma	Retorna el numero del color de la pluma	Operador
BG	Retorna el numero del color del fondo	Operador
SETPOS lista	Coloca la tortuga en la posicion dada por las coordenadas	Comando
POS	Retorna una lista donde contiene las coordenadas del punto donde se encuentra la tortuga	Operador
XCOR	Retorna el valor de la coordenada X del punto	Operador
YCOR	Retorna el valor de la coordenada Y del punto	Operador
SETX numero	Coloca a la tortuga en el punto de la coordenada X sin variar su coordenada Y	Comando
SETY numero	Coloca a la tortuga en el punto de la coordenada Y sin variar su coordenada X	Comando
SETH numero	Coloca a la tortuga en la posicion dada por el numero	Comando
HEADING	Retorna la orientacion de la tortuga	Operador
SETSP numero	Hace que la tortuga se mantenga en movimiento a la velocidad dada por el numero	Comando
SPEED	Retorna la velocidad actual de la tortuga	Operador

TELL n. tortuga	Define que tortugas seran las actuales en ese momento	Comando
WHO	Da una lista de las tortugas con el numero en curso	Operador
COLOR	Da el numero del color de la tortuga en curso	Operador

TABLA DE COMANDOS

Instruccion	Comentarios	Operador/comando
SUM n1 n2	Suma los datos de entrada	Operador
n1 + n2	Suma los datos de entrada (infijo)	Operador
PRODUCT n1 n2	Multiplica los datos de entrada	Operador
n1 * n2	Multiplica los datos de entrada (infijo)	Operador
n1 - n2	Resta de los datos de entrada (infijo)	Operador
n1 / n2	Divide los datos de entrada (infijo)	Operador
SQRT n1	Calcula la raiz cuadrada del dato de entrada	Operador
SIN n1	Calcula el seno del dato	Operador
COS n1	Calcula el coseno del dato	Operador
RANDOM n1	Calcula un numero aleatorio inferior al valor dado	Operador
RERANDOM	Recuerda la ultima secuencia generada por numeros aleatorios	Comando
n1 > n2	Da el dato logico verdadero si n1 es mayor que n2	Operador
n1 < n2	Da el dato logico verdadero si n1 es menor que n2	Operador
EQUALP n1 n2	Da el dato logico verdadero si n1 y n2 son iguales	Operador
n1 = n2	Da el dato logico verdadero si n1 y n2 son iguales (infijo)	Operador
AND	Realiza la funcion logica AND de los datos de entrada	Operador
OR	Realiza la funcion logica OR de los datos de entrada	Operador
NOT	Da el valor logico opuesto al dato de entrada	Operador

USO DEL DISCO DEL AMSTRAD

ISBN.: 84-86586-01-01
Depósito Legal: M. 27.568-1986
Autor: C. LONGHI
Edita: G.T.S., S. A. - C/. Bailén, 20, 1.° izda. - 28005 Madrid.
Imprime: Gráficas FUTURA, Sdad. Coop. Ltda. - Villafranca del Bierzo, 21-23. Fuenlabrada
Distribuye: R.B.A., Promotora de Ediciones, S. A. - Travesera de Gracia, 56. Atico 1.ª. Teléfono: (93) 200 82 56.
Prohibida la reproducción de textos, parcial o totalmente, ni aún citando su procedencia.

OTROS TITULOS:

* **MS-DOS**
  por Antonio Bellido

* **LOS TRUCOS DEL AMSTRAD**
  por Antonio Bellido.

* **LA BIBLIA DEL AMSTRAD**
  por Christian Lomghi

P.V.P.: 600 PTAS.

Edita: Editorial GTS.
C/. Bailén, 20. 1.º Izda. 28005 MADRID.
Depósito Legal: M-27569-1986
I.S.B.N.: 84-86586-01-1
Imprime Gráficas FUTURA. Sdad. Coop. Ltda.



[FRA] Ce document a été préservé numériquement à des fins éducatives et d'études, et non commerciales.
[ENG] This document has been digitally preserved for educational and study purposes, not for commercial purposes.
[ESP] Este documento se ha conservado digitalmente con fines educativos y de estudio, no con fines comerciales.